



# Thuyết minh đồ án tốt nghiệp Cơ điện tử

Kỹ thuật phần mềm ứng dụng (Trường Đại học Bách khoa Hà Nội)

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**



**TRƯỜNG CƠ KHÍ  
BỘ MÔN CƠ ĐIỆN TỬ**



## **ĐỒ ÁN TỐT NGHIỆP**

### **Thiết kế mô hình nhà thông minh**

**TRẦN LONG QUANG ANH**

anh.tlp184041@sis.hust.edu.vn

**NGUYỄN TRUNG HIẾU**

hieu.nt184446@sis.hust.edu.vn

**TRẦN TRỌNG SINH**

sinh.tt184604@sis.hust.edu.vn

**Chuyên ngành Cơ Điện Tử**

**Giảng viên hướng dẫn:** TS. Phạm Đức An

\_\_\_\_\_  
Chữ ký của GVHD

**Bộ môn:** Cơ điện tử

**Trường:** Cơ khí

**HÀ NỘI, 8/2022**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT  
NAM  
Độc lập – Tự do – Hạnh phúc**

**NHIỆM VỤ  
THIẾT KẾ ĐỒ ÁN TỐT NGHIỆP**

Họ và tên sinh viên: Trần Long Quang Anh	MSSV: 20184041
Nguyễn Trung Hiếu	MSSV: 20184446
Trần Trọng Sinh	MSSV: 20184602

Lớp: CTTN Cơ điện tử - K63

Khoa: Cơ Điện Tử

Trường: Cơ Khí

**I/ ĐỀ TÀI THIẾT KẾ**

**“Thiết kế mô hình nhà thông minh”**

**II/ CÁC SỐ LIỆU BAN ĐẦU**

**III/ NỘI DUNG THUYẾT MINH VÀ TÍNH TOÁN**

- Tổng quan về đề tài
- Thiết kế mô hình cho đề tài
- Nghiên cứu dịch vụ Firebase ứng dụng trong truyền nhận dữ liệu của hệ thống IoT
- Nghiên cứu giao thức truyền nhận dữ liệu HTTP và MQTT
- Hệ thống nhận diện khuôn mặt
- Nghiên cứu phát triển App Desktop
- Nghiên cứu phát triển App Android
- Kết quả
- Kết luận

**IV/ CÁC BẢN VẼ VÀ ĐỒ THỊ**

**V/ CÁN BỘ HƯỚNG DẪN: TS. Phạm Đức An**

**VI/ NGÀY GIAO NHIỆM VỤ THIẾT KẾ: 04/2022**

**VII/ NGÀY HOÀN THÀNH ĐỒ ÁN: 08/2022**

Hà Nội, ngày      tháng      năm 2022  
Giảng viên hướng dẫn



### Đánh giá của giảng viên hướng dẫn

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

## Kết quả đánh giá

Hà Nội, ngày    tháng    năm 2022

Họ và tên	Điểm
Trần Long Quang Anh	
Nguyễn Trung Hiếu	
Trần Trọng Sinh	

Giảng viên hướng dẫn

[illegible]

Hà Nội, ngày    tháng    năm 2022

Họ và tên	Điểm
Trần Long Quang Anh	
Nguyễn Trung Hiếu	
Trần Trọng Sinh	

Giảng viên phản biện

### Lời cảm ơn

Sau 4 năm học tập, rèn luyện tại trường Đại học Bách Khoa Hà Nội, nhận được sự giảng dạy nhiệt tình từ các thầy cô, đặc biệt là các thầy cô trường Cơ Khí – Đại học Bách Khoa Hà Nội, chúng em đã quyết định lựa chọn đề tài **“Thiết kế mô hình nhà thông minh”** cho đồ án tốt nghiệp của mình.

Chúng em xin chân thành cảm ơn TS. Phạm Đức An cùng các thầy cô bộ môn đã hướng dẫn, chỉ bảo tận tình, giúp chúng em hoàn thành đề tài kịp tiến độ. Là những sinh viên năm 4, vốn kiến thức còn ít ỏi và kinh nghiệm thực tế lại chưa nhiều nên trong quá trình làm đồ án không thể tránh khỏi được những thiếu sót. Chúng em rất mong nhận được những phản hồi, đóng góp sửa chữa để đề tài của chúng em hoàn thiện hơn. Chúng em cảm ơn.

### Tóm tắt nội dung đồ án

Đồ án là kết quả học tập, nghiên cứu tìm tòi của nhóm khi thực hiện đề tài **“Thiết kế mô hình nhà thông minh”**. Vấn đề được đặt ra ban đầu là tìm hiểu, nghiên cứu về các mô hình nhà thông minh đang được sử dụng tại Việt Nam và trên thế giới. Đề tài đem tới cái nhìn tổng quan về quy trình hoạt động của một hệ thống IoT nói chung và hệ thống nhà thông minh – Smart Home nói riêng. Đa số các hệ thống nhà thông minh hiện nay giá thành tương đối cao, chưa kể tới quy trình thi công tương đối phức tạp nên mô hình thực tế **QASHome** của nhóm đem tới trải nghiệm thực tế trong việc quản lý một ngôi nhà thông minh. Với những kết quả đạt được, hoàn toàn phù hợp với những mục tiêu đã đề ra, đồng thời đồ án có thể là cơ sở cho việc thi công các mô hình nhà thông minh khác trong thực tế. Ngoài ra, mô hình nhà thông minh của nhóm trong tương lai có thể được phát triển theo các hướng hệ thống IoT mở rộng. Nhờ đồ án, chúng em đã học được thêm nhiều kiến thức cũng như các kỹ năng mới như xây dựng một hệ thống cơ điện tử ứng dụng IoT, các kỹ năng lập trình mới, có cái nhìn tổng quan nhất về quy trình xây dựng một hệ thống cơ điện và đặc biệt là tinh thần làm việc nhóm.

## MỤC LỤC

Lời nói đầu	1
Phần mở đầu	2
1.1. Đặt vấn đề	2
1.2. Mục đích nghiên cứu	2
1.3. Phương pháp nghiên cứu	2
CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI	3
1.1. Tổng quan về nhà thông minh	3
1.1.1. Bối cảnh và nhu cầu sử dụng hệ thống nhà thông minh	4
1.1.2. Các mô hình nhà thông minh được áp dụng hiện nay	5
1.2. Nghiên cứu các tính năng nổi bật trong nhà thông minh hiện nay	6
CHƯƠNG 2. THIẾT KẾ MÔ HÌNH CHO ĐỀ TÀI	8
2.1. Lựa chọn hướng thiết kế cho đề tài	8
2.2. Tính năng và yêu cầu đặt ra cho mô hình nhà thông minh của đề tài	8
2.3. Các thành phần cơ bản trong nhà thông minh	9
2.3.1. Hệ thống chiếu sáng	9
2.3.2. Hệ thống kiểm soát ra vào	9
2.3.3. Hệ thống cảnh báo rò rỉ khí gas	9
2.3.4. Hệ thống kiểm soát môi trường	9
2.3.5. Hệ thống giếng trời	9
2.3.6. Hệ thống kiểm tra, điều khiển, giám sát từ xa	20
CHƯƠNG 3. NGHIÊN CỨU DỊCH VỤ FIREBASE ỨNG DỤNG TRONG TRUYỀN NHẬN DỮ LIỆU CỦA HỆ THỐNG IoT	21
3.1. Giới thiệu về nền tảng Firebase	21
3.1.1. Các dịch vụ nổi bật của Firebase	21
3.1.2. Một số ứng dụng sử dụng nền tảng Firebase	21
3.1.3. Ưu, nhược điểm khi sử dụng Firebase	22
3.2. Giới thiệu về dịch vụ Firebase Realtime Database	22
3.2.1. Các khả năng chính của Firebase Realtime Database	22
3.2.2. Lưu trữ dữ liệu trong Firebase Realtime Database	22
3.2.3. Thực hiện các thao tác CRUD (Creat – Read – Update – Delete)	23
3.3. Ứng dụng của Firebase Realtime Database vào đề tài nhà thông minh	24



CHƯƠNG 4.    NGHIÊN CỨU GIAO THỨC TRUYỀN NHẬN DỮ LIỆU HTTP VÀ MQTT	30
4.1. Nghiên cứu sử dụng giao thức HTTP truyền nhận dữ liệu thông tin giữa app và bộ điều khiển chức năng nhà thông minh	30
4.1.1. Giới thiệu giao thức HTTP	30
4.1.2. Các phương thức sử dụng trong giao thức HTTP	30
4.1.3. Ứng dụng giao thức HTTP trong đề tài	31
4.2. Nghiên cứu sử dụng giao thức MQTT truyền nhận dữ liệu thông tin giữa app android và Firebase Realtime Database	34
4.2.1. Giới thiệu giao thức MQTT	35
4.2.2. Nguyên lý làm việc của giao thức MQTT	35
4.2.3. Ưu nhược điểm của giao thức MQTT	35
4.2.4. Ứng dụng giao thức MQTT trong đề tài	36
4.3. Mã hoá dữ liệu	36
4.3.1. Encryption và Decryption	36
4.3.2. Mã hoá dữ liệu SHA – 512	40
CHƯƠNG 5.    THIẾT KẾ MÔ HÌNH NHÀ THÔNG MINH	45
5.1. Sơ đồ kết cấu của ngôi nhà	45
5.2. Nguyên lý hoạt động của mô hình nhà thông minh	45
5.2.1. Sơ đồ nguyên lý chung	46
5.2.2. Sơ đồ khối hệ thống và chức năng	49
5.3. Lựa chọn linh kiện và module trong mạch điện tử	49
5.3.1. Module vi điều khiển Arduino Mega 2560	49
5.3.2. Module cảm biến nhiệt độ, độ ẩm DHT11	50
5.3.3. Relay 1 kênh 5V kích mức thấp	52
5.3.4. Module cảm biến hồng ngoại IR	53
5.3.5. Module cảm biến nồng độ khí gas MQ – 6	54
5.3.6. Module cảm biến mưa	55
5.3.7. Module Sim – 800L	56
5.3.8. Thiết bị ngoại vi quạt	58
5.3.9. Thiết bị ngoại vi Servo SG – 90	59
5.3.10. Module mạch giảm áp LM2596	59
5.4. Mạch điện trong các khu vực phòng	60
5.4.1. Mạch điện phòng khách	61
5.4.2. Mạch điện cầu thang	61

5.4.3. Mạch điện phòng bếp	62
5.4.4. Mạch điện phòng ngủ 1, 2	62
5.4.5. Mạch điện khu vực giếng trời	63
<b>CHƯƠNG 6. HỆ THỐNG NHẬN DIỆN KHUÔN MẶT</b>	<b>64</b>
6.1. Tổng quan hệ thống	64
6.2. Mô hình thuật toán MTCNN	64
6.2.1. Stage 1: P-Net (Proposal Network)	65
6.2.2. Stage 2: R-Net (Refine Network)	66
6.2.3. Stage 3: O-Net (Output Network)	67
6.3. Mô hình thuật toán FaceNet	67
6.3.1. Deep Convolutional Networks	67
6.3.2. Triple loss	68
6.4. Thuật toán SVM	69
6.4.1. Thuật toán SVM	69
6.4.2. Tối ưu SVM	71
6.5. Ứng dụng hệ thống nhận diện khuôn mặt trong đề tài	73
<b>CHƯƠNG 7. NGHIÊN CỨU PHÁT TRIỂN DESKTOP APP</b>	<b>75</b>
7.1. Công cụ và ngôn ngữ lập trình	75
7.1.1. Cấu trúc phần mềm	75
7.1.2. Quy trình xây dựng phần mềm	75
7.2. Ứng dụng dịch vụ Firebase trong phần mềm	76
7.3. Xây dựng giao diện Desktop App	76
<b>CHƯƠNG 8. NGHIÊN CỨU PHÁT TRIỂN APP ANDROID</b>	<b>78</b>
8.1. Công cụ và ngôn ngữ lập trình	78
8.1.1. Cấu trúc dự án	78
8.1.2. Giao diện người dùng trên Android Studio	79
8.1.3. Quy trình các bước lập trình trên Android Studio	80
8.2. Ứng dụng dịch vụ Firebase trong Android Studio	82
8.3. Xây dựng giao diện Android App	82
<b>CHƯƠNG 9. KẾT QUẢ</b>	<b>85</b>
9.1. Mô hình nhà thông minh	85
9.2. Giao diện Desktop App	85
9.3. Giao diện Android App	90
9.4. Hệ thống nhận diện khuôn mặt	95

9.5. Mô hình giếng trời	96
CHƯƠNG 10. KẾT LUẬN	97
10.1. Kết luận	97
10.2. Hướng phát triển của đồ án trong tương lai	97
TÀI LIỆU THAM KHẢO	98
PHỤ LỤC	100

## DANH MỤC HÌNH

Hình 1. 1. Hệ thống nhà thông minh hiện nay.....	3
Hình 1. 2. Hệ thống nhà thông minh quản lý chiếu sáng gia đình.....	4
Hình 1. 3. Hệ thống nhà thông minh Lumi.....	5
Hình 1. 4. Hệ thống nhà thông minh Vsmart.....	6
Y	
Hình 2. 1. Mô hình chung của nhà thông minh.....	8
Hình 2. 2. Mô hình mái và ray 1.....	10
Hình 2. 3. Mô hình má và ray 2.....	10
Hình 2. 4. Mô hình thiết kế ban đầu.....	10
Hình 2. 5. Một cánh cửa mái.....	11
Hình 2. 6. Mô hình mái đầy đủ.....	11
Hình 2. 7. Thiết kế mô hình khung dạng khác.....	13
Hình 2. 8. Mở giếng trời, mái thu.....	13
Hình 2. 9. Đóng giếng trời, mái xếp.....	13
Hình 2. 10. Mái che giếng trời.....	14
Hình 2. 11. Bộ phận nâng mái.....	14
Hình 2. 12. Động cơ.....	14
Hình 2. 13. Hộp gắn ren.....	15
Hình 2. 14. Cụm thanh quay và hộp gắn ren.....	15
Hình 2. 15. Thanh ray.....	15
Hình 2. 16. Bánh ray ăn khớp với thanh ray.....	15
Hình 2. 17. Đóng giếng trời.....	16
Hình 2. 18. Mở giếng trời.....	16
Hình 2. 19. Mái che giếng trời.....	16
Hình 2. 20. Bộ phận nâng mái.....	17
Hình 2. 21. Bộ phận dẫn hướng.....	17
Hình 2. 22. Các thông số của hệ mái và ray dẫn.....	18
Hình 2. 23. Phương và chiều của lực và momen tác động lên trục động cơ.....	18
Hình 2. 24. Datasheet KGC-3448 SERIES.....	19
Hình 3. 1. Ví dụ về một chuỗi JSON.....	23
Hình 3. 2. Tham chiếu tới cơ sở dữ liệu JSON.....	23
Hình 3. 3. Minh họa với thao tác Insert data.....	23
Hình 3. 4. Minh họa với thao tác Reading data.....	24
Hình 3. 5. Minh họa với thao tác Updating data.....	24
Hình 3. 6. Cấu trúc quản lý dữ liệu của dự án.....	25

Hình 3. 7. Quản lý dữ liệu của phòng khách.....	25
Hình 3. 8. Quản lý dữ liệu của phòng bếp.....	26
Hình 3. 9. Quản lý dữ liệu của phòng ngủ 1.....	26
Hình 3. 10. Quản lý dữ liệu của phòng ngủ 2.....	26
Hình 3. 11. Quản lý dữ liệu phòng tắm.....	<b>Error! Bookmark not defined.</b>
Hình 3. 12. Quản lý dữ liệu khu vực giếng trời.....	27
Hình 3. 13. Quản lý dữ liệu về điều kiện khu vực.....	27
Hình 3. 14. Quản lý dữ liệu về thông tin đăng nhập người dùng.....	28
Hình 3. 15. Quản lý dữ liệu đăng ký người dùng.....	29
Hình 3. 16. Quản lý dữ liệu khi muốn lấy lại thông tin đã quên.....	29
Hình 4. 1. Giao thức truyền tải HTTP.....	30
Hình 4. 2. Cấu trúc tín hiệu input.....	32
Hình 4. 3. Cấu trúc tín hiệu output.....	34
Hình 4. 4. Nguyên lý làm việc của giao thức MQTT.....	35
Hình 4. 5. Mô hình ứng dụng giao thức MQTT trong đề tài.....	36
Hình 4. 6. Reverse Cipher.....	37
Hình 4. 7. Mã hóa Ceasar Cipher.....	37
Hình 4. 8. Mã hóa ROT13.....	38
Hình 4. 9. Bảng mã hóa.....	38
Hình 4. 10. Bảng mã hóa.....	38
Hình 4. 11. Bảng quy đổi.....	39
Hình 4. 12. Thuật toán Affine Cipher trong thư viện Cryptography sẽ kết hợp thuật toán Multiplicative Cipher và Caesar Cipher.....	<b>Error! Bookmark not defined.</b>
Hình 4. 13. Thuật toán Affine Cipher.....	40
Hình 4. 14. Sơ đồ thuật toán.....	41
Hình 4. 15. Sơ đồ khối tổng thể.....	41
Hình 4. 16. Quá trình xử lý mỗi block 1024-bit.....	43
Hình 4. 17. Sơ đồ hoạt động mỗi round.....	44
Hình 5. 1. Sơ đồ mô hình nhà thông minh.....	45
Hình 5. 2. Trường hợp người dùng phòng khách.....	46
Hình 5. 3. Trường hợp người dùng khu vực cầu thang.....	47
Hình 5. 4. Trường hợp người dùng phòng bếp.....	47
Hình 5. 5. Trường hợp người dùng các phòng ngủ.....	48
Hình 5. 6. Trường hợp người dùng khu vực giếng trời.....	48
Hình 5. 7. Sơ đồ khối hoạt động của toàn bộ mô hình.....	49

Hình 5. 8. Vi điều khiển Arduino Mega 2560.....	49
Hình 5. 9. Cảm biến DHT11.....	51
Hình 5. 10. Module cảm biến DHT11 được nhóm sử dụng.....	51
Hình 5. 11. Relay 1 kênh 5V kích mức thấp.....	52
Hình 5. 12. Module cảm biến hồng ngoại IR.....	53
Hình 5. 13. Module cảm biến hồng ngoại được nhóm sử dụng.....	53
Hình 5. 14. Module cảm biến nồng độ khí gas MQ-6.....	54
Hình 5. 15. Module cảm biến mưa.....	55
Hình 5. 16. Module nghe gọi nhấn tin Sim - 800L.....	56
Hình 5. 17. Sơ đồ chân của module Sim - 800L.....	56
Hình 5. 18. Thiết bị ngoại vi quạt 12V.....	58
Hình 5. 19. Động cơ Servo SG90.....	59
Hình 5. 20. Module mạch giảm áp LM2596.....	60
Hình 5. 21. Sơ đồ nối dây phòng khách.....	61
Hình 5. 22. Sơ đồ nối dây cầu thang.....	61
Hình 5. 23. Sơ đồ nối dây phòng bếp.....	62
Hình 5. 24. Sơ đồ nối dây các phòng ngủ.....	62
Hình 5. 25. Sơ đồ nối dây khu vực giếng trời.....	63
Hình 6. 1. Mô hình thuật toán MTCNN.....	64
Hình 6. 2. Mạng P-Net.....	65
Hình 6. 3. Resize ảnh.....	65
Hình 6. 4. Thuật toán Non-Maximum Suppression.....	66
Hình 6. 5. Minh hoạ Stage 1 P-Net.....	66
Hình 6. 6. Mạng R-Net.....	66
Hình 6. 7. Minh hoạ Stage 1 R-Net.....	67
Hình 6. 8. Mạng O-Net.....	67
Hình 6. 9. Minh hoạ Stage 2 O-Net.....	67
Hình 6. 10. Kiến trúc mạng GoogLeNet – Inception v1.....	68
Hình 6. 11. Triplet Loss.....	69
Hình 6. 12. Minh hoạ phân lớp dữ liệu.....	70
Hình 6. 13. Xác định hyper plane.....	70
Hình 6. 14. Tối ưu hyper plane.....	71
Hình 6. 15. Phân lớp bộ dữ liệu.....	71
Hình 6. 16. Tối ưu margin.....	72
Hình 6. 17. Tiền xử lý bộ dữ liệu.....	73
Hình 6. 18. Quá trình training.....	74

Hình 6. 19. Quá trình nhận diện.....	74
Hình 7. 1. 04 class sử dụng để xây dựng Desktop App.....	75
Hình 7. 2. Sơ đồ quy trình thiết kế phần mềm.....	75
Hình 8. 1. Các tệp dự án trong chế độ xem “Android”.....	78
Hình 8. 2. Giao diện người dùng trên Android Studio.....	79
Hình 8. 3. Quy trình xây dựng giao diện Android App.....	80
Hình 8. 4. File.xml trong tool windows.....	80
Hình 8. 5. Giao diện file.xml.....	81
Hình 8. 6. File.java trên Android Studio.....	82
Hình 8. 7. Ví dụ về xây dựng giao diện phòng khách.....	83
Hình 8. 8. Ví dụ xây dựng giao diện phòng khách trên file.java.....	84
Hình 9. 1. Mô hình nhà thông minh thực tế.....	85
Hình 9. 2. Giao diện đăng nhập của ứng dụng Desktop App.....	85
Hình 9. 3. Giao diện đăng ký của ứng dụng.....	86
Hình 9. 4. Nếu người dùng quên tài khoản mật khẩu.....	86
Hình 9. 5. Giao diện "Home" của Desktop App.....	87
Hình 9. 6. Giao diện quản lý Living room của Desktop App.....	87
Hình 9. 7. Giao diện quản lý Kitchen của Desktop App.....	88
Hình 9. 8. Giao diện quản lý Bedroom 1 của Desktop App.....	88
Hình 9. 9. Giao diện quản lý Bedroom 2 của Desktop App.....	89
Hình 9. 10. Giao diện quản lý khu vực giếng trời của Desktop App.....	89
Hình 9. 11. Giao diện quản lý điều kiện sống Life Condition.....	90
Hình 9. 12. Giao diện đăng nhập của ứng dụng.....	90
Hình 9. 13. Giao diện chính của ứng dụng.....	91
Hình 9. 14. Giao diện quản lý phòng khách.....	92
Hình 9. 15. Giao diện quản lý phòng ngủ 1.....	92
Hình 9. 16. Giao diện quản lý phòng ngủ 2.....	93
Hình 9. 17. Giao diện quản lý phòng bếp.....	94
Hình 9. 18. Giao diện quản lý khu vực giếng trời.....	95
Hình 9. 19. Phân biệt nhiều người trong cùng khung hình.....	95
Hình 9. 20. Nhận diện chính xác người dùng.....	96

## DANH MỤC BẢNG BIỂU

Bảng 1. Bảng Danevit - Hatenberg.....	<b>Error! Bookmark not defined.</b>
Bảng 2. Sơ đồ nối chân của DHT11.....	50
Bảng 3. Thông số kỹ thuật của Relay sử dụng.....	52



## Lời nói đầu

Từ xưa đến nay, nhân loại đã không ngừng học hỏi, tìm tòi và nghiên cứu để tạo ra những sản phẩm nhằm phục vụ cho sinh hoạt hằng ngày, những sản phẩm ấy luôn phát triển theo hướng tự động hoá và ngày càng thông minh để có thể phục vụ những nhu cầu ngày càng cao của con người. Với xu hướng này, nhiều nước trên thế giới đã ứng dụng khoa học kỹ thuật hiện đại để những thiết bị hiện đại trong gia đình hoạt động, điều khiển tự động. Với sự ra đời của các hệ thống IoT khi mà mọi thiết bị có thể được kết nối với nhau đã giúp cho con người có thể xây dựng được những căn nhà thông minh có thể điều khiển một cách dễ dàng thông qua các sản phẩm thông minh. Hiện nay ở Việt Nam đã có khá nhiều tập đoàn, công ty nghiên cứu về lĩnh vực này, đây có thể sẽ là một hướng đi tiềm năng trong những năm tới.

Với mong muốn được học tập, nghiên cứu và vận dụng kiến thức đã được học trên giảng đường. Nhóm chúng em đã lựa chọn đề tài “**Thiết kế mô hình nhà thông minh**”.

Trong quá trình thực hiện đề tài đồ án tốt nghiệp này, chúng em xin gửi lời cảm ơn chân thành tới TS. Phạm Đức An, trường Cơ Khí, Đại học Bách Khoa Hà Nội, đã hướng dẫn tận tình và chỉ dẫn các bước, cung cấp những tài liệu nghiên cứu quý báu, hướng nghiên cứu để chúng em có thể thực hiện được các yêu cầu của đồ án.

Trong quá trình thực hiện đề tài, dựa theo những kết quả đạt được bước đầu, dù đã rất cố gắng tuy nhiên không thể tránh khỏi được những thiếu sót và hạn chế nhất định. Vì vậy, chúng em rất mong nhận được sự góp ý, bổ sung của các thầy cô để đề tài được tối ưu và hoàn thiện hơn.

## **Phần mở đầu**

### **1.1. Đặt vấn đề**

Hiện nay không chỉ trên thế giới mà tại Việt Nam dưới sự bùng nổ của cách mạng công nghệ thông tin đã giúp cho đời sống của con người ngày càng nâng cao. Các thiết bị tự động hoá đã ngày càng chiếm vai trò quan trọng trong nền sản xuất và thậm chí là vào cuộc sống sinh hoạt thường ngày của con người. Do đó, một ngôi nhà thông minh không còn là mơ ước viễn vông của con người mà đó đã được thực hiện hoá. Qua báo chí, phương tiện truyền thông, Internet chúng ta có thể dễ dàng tìm thấy những mô hình nhà thông minh đẹp mắt, hữu ích lại vô cùng thân thiện.

Nhà thông minh được xây dựng trên nền tảng của công nghệ IoT, với giá thành đầu tư cho hệ thống không quá đắt, phù hợp với điều kiện kinh tế của các gia đình tại Việt Nam. Mặc dù hệ thống chưa được phát triển mạnh mẽ, sâu rộng song hứa hẹn nhiều kết quả tốt đẹp trong tương lai.

Là một sinh viên khoa Cơ điện tử, trường Cơ Khí, Đại học Bách Khoa Hà Nội, với những kiến thức được học cùng mong muốn thiết kế một ngôi nhà thông minh có thể đáp ứng được các nhu cầu sinh hoạt thường ngày. Dưới sự hướng dẫn của TS. Phạm Đức An, chúng em lựa chọn đề tài “Thiết kế mô hình nhà thông minh” làm đề tài cho đồ án tốt nghiệp.

### **1.2. Mục đích nghiên cứu**

Đề tài được nghiên cứu, khảo sát và thực hiện với mục đích áp dụng các kiến thức đã được học trên ghế nhà trường để xây dựng, phát triển một mô hình nhà thông minh. Sau khi thực hiện, đề tài đưa ra được các mục đích sau:

- Tìm hiểu, xây dựng một hệ thống toàn diện có tính ứng dụng thực tiễn
- Phần cứng có thể dễ dàng triển khai, giảm thiểu tối đa chi phí phát sinh

### **1.3. Phương pháp nghiên cứu**

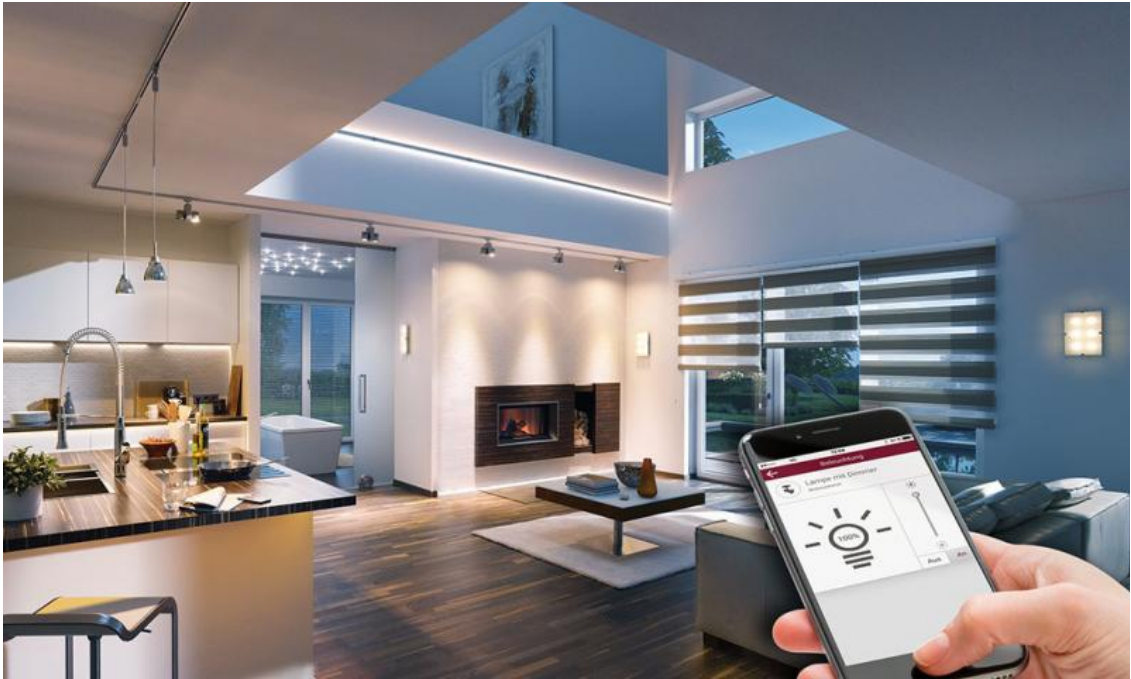
Trong đề tài này, nhóm đã sử dụng các phương pháp nghiên cứu:

- Phương pháp tham khảo tài liệu: bằng cách thu thập thông tin từ sách, tạp chí, các nguồn thông tin uy tín trên internet
- Phương pháp quan sát: khảo sát một số ứng dụng có sử dụng công nghệ IoT, các hệ thống nhà thông minh trên thế giới và tại Việt Nam
- Phương pháp thực nghiệm: Xem xét một số công nghệ đã được áp dụng trước đó để rút ra kinh nghiệm cũng như những yêu cầu đề ra cho hệ thống mô hình nhà thông minh

## CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

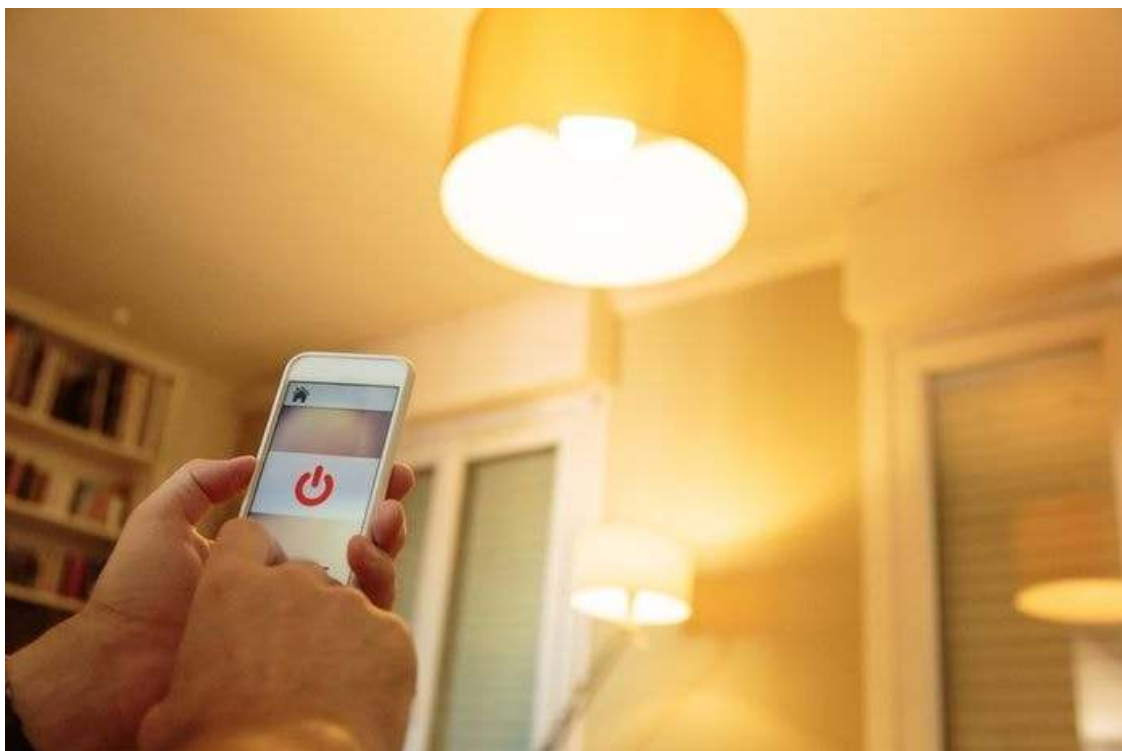
### 1.1. Tổng quan về nhà thông minh

Nhà thông minh (Smart Home) là căn nhà được tích hợp các thiết bị, điện tử được điều khiển, giám sát, truy cập từ xa. Khác với nhà tự động (Home Automation) đơn thuần chỉ là các thiết bị hoạt động theo lịch trình cài đặt sẵn, Smart Home được xây dựng trên nền tảng khái niệm Internet vạn vật (IoT). Tại đây, các thiết bị trong nhà có thể trao đổi thông tin với nhau, điều chỉnh các chức năng theo thói quen sinh hoạt hàng ngày của người dùng.



*Hình 1. 1. Hệ thống nhà thông minh hiện nay*

Một trong những ví dụ cơ bản nhất của nhà thông minh là một hệ thống kiểm soát mức độ chiếu sáng của hệ thống đèn giúp tiết kiệm điện và phù hợp với nhu cầu người dùng. Chẳng hạn như việc cài đặt đèn ánh sáng nhẹ cho các bữa tiệc tối, đèn dịu cho người dùng đọc sách, đèn ngủ... Hệ thống cũng có thể điều chỉnh rèm cửa theo yêu cầu, kiểm soát nhiệt độ, hệ thống camera giám sát, hệ thống khoá cửa tự động, hệ thống phòng ngừa trộm.



*Hình 1. 2. Hệ thống nhà thông minh quản lý chiếu sáng gia đình*

### **1.1.1. Bối cảnh và nhu cầu sử dụng hệ thống nhà thông minh**

Ngày nay, khi đời sống ngày càng được nâng cao, nhu cầu của con người đòi hỏi những sự tiện nghi và hỗ trợ tốt nhất. Cùng với đó là sự mở rộng không ngừng của mạng lưới internet trên khắp các vùng quốc gia và lãnh thổ làm cho việc giám sát và điều khiển hệ thống qua mạng internet trở thành tất yếu. Từ những yêu cầu và điều kiện thực tế đó, ý tưởng về ngôi nhà thông minh được hình thành, nơi mà mọi hoạt động của con người đều được hỗ trợ và giúp đỡ một cách linh hoạt, ngoài ra ngôi nhà còn có thể tự động quản lý một cách thông minh nhất. Vậy, như thế nào là nhà thông minh? Sự thông minh của một ngôi nhà được thể hiện trên 4 phương diện như sau:

- Thứ nhất, là khả năng tự động hóa. Căn nhà được trang bị hệ thống các cảm biến như: cảm biến nhiệt độ, cảm biến độ ẩm, cảm biến khí gas, cảm biến báo cháy, cảm biến vật cản, cảm biến ánh sáng... với khả năng tự động hoạt động theo điều kiện môi trường. Nhà thông minh giúp chúng ta giám sát được mức tiêu thụ điện, nước tốt hơn so với thông thường.
- Thứ hai, là khả năng thỏa mãn nhu cầu của người sử dụng. Chủ nhân ngôi nhà có thể điều khiển theo ý muốn hoặc theo những kịch bản được lập trình sẵn.
- Thứ ba, là khả năng bảo mật, giám sát an ninh. Hệ thống giám sát an ninh, báo cháy, báo rò rỉ khí gas sẽ tự động báo trạng thái của ngôi nhà qua mạng internet.
- Thứ tư, là khả năng điều khiển, cảnh báo từ xa thông qua kết nối internet thông qua wifi, 3G... Các thiết bị như: bóng đèn, điều hòa, ti vi, tủ lạnh,... cũng đều được kết nối tới mạng internet. Người sử dụng chỉ cần có một thiết bị kết nối internet là có thể theo dõi dữ liệu từ các cảm biến và điều khiển các thiết bị trong nhà theo ý muốn của bản thân.

### 1.1.2. Các mô hình nhà thông minh được áp dụng hiện nay

Trên thị trường Việt Nam hiện có khá nhiều nhà cung cấp các giải pháp “Nhà thông minh”:

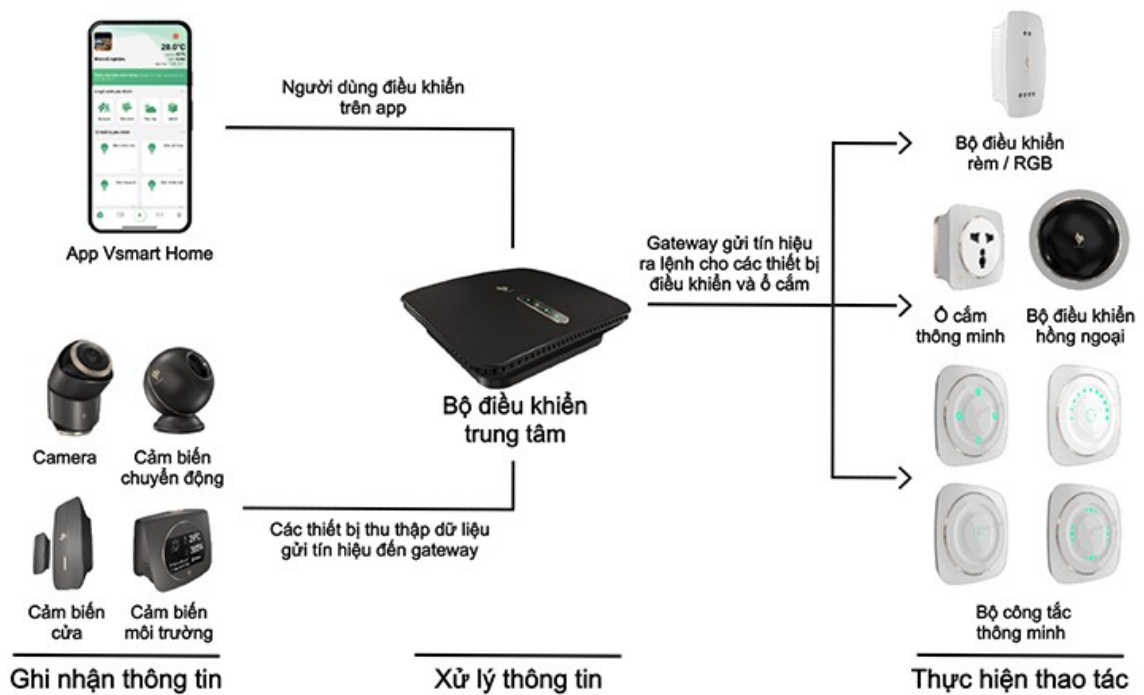
- **Nhà thông minh Bkav** kết nối các thiết bị trong nhà như đèn chiếu sáng, rèm cửa, điều hòa nhiệt độ, tivi, âm thanh, khóa cửa, bình nóng lạnh, quạt thông gió, camera an ninh,... thành hệ thống mạng, người dùng dễ dàng kiểm soát ngôi nhà thông qua một giao diện trực quan 3D trên điện thoại hay máy tính bảng và có thể kiểm soát ngôi nhà từ bất kỳ đâu.

- **Nhà thông minh Lumi:** Sản xuất và cung cấp các thiết bị thông minh như chiết áp cảm ứng, cảm biến, bộ điều khiển hồng ngoại, công tắc cảm ứng, bộ điều khiển trung tâm,... Các lợi ích của giải pháp Nhà thông minh Lumi Smarthome.



*Hình 1. 3. Hệ thống nhà thông minh Lumi*

- **Nhà thông minh Vsmart :** phát triển cả phần cứng và phần mềm quản lý với hệ thống đèn điện, điều hòa, tivi, bình nóng lạnh, rèm cửa... bằng smartphone và giọng nói.



Hình 1. 4. Hệ thống nhà thông minh Vsmart

- **Nhà thông minh Acis:** cung cấp các giải pháp hệ thống ánh sáng thông minh, điều khiển điều hòa TV thông minh, rèm tự động, an ninh báo trộm, khóa cửa thông minh, âm thanh giải trí,... giúp hiện đại hóa không gian sống trở nên tiện nghi hơn, đón đầu xu hướng công nghệ của thời đại mới.

- **Nhà thông minh Xiaomi:** tất cả mọi thứ được điều khiển trên app Mi Home và tích hợp được với cả Google Home và Alexa, chủ nhà có thể điều khiển thao tác trên điện thoại hoặc qua giọng nói. Một số sản phẩm smarthome của Xiaomi như máy lọc không khí; robot quét nhà; máy lọc nước; ổ cắm thông minh, công tắc gắn tường thông minh; nồi cơm điện thông minh; đèn bàn, đèn ngủ thông minh; camera an ninh; đèn ốp trần nhà; các loại cảm biến cửa, cảm biến nhiệt độ, cảm biến hồng ngoại,...

- **Nhà thông minh Google Home:** giải pháp điều khiển tất cả các thiết bị điện thông qua một chiếc loa với nhiều tính năng vượt trội, điều khiển ngôi nhà bằng giọng nói. Các lợi ích của giải pháp Nhà thông minh Google Home: gọi điện mà không phải dùng đến tay; chuyển chế độ ban ngày sang ban đêm; điều khiển các thiết bị điện trong ngôi nhà chỉ bằng giọng nói; cập nhật tin tức hàng ngày ...

Ngoài ra trên thị trường còn có các giải pháp Nhà thông minh đến từ các thương hiệu lớn của nước ngoài như Siemens (Đức), Schneider (Pháp), Smart G4 (Mỹ), Gamma (Đức), Fibaro (Ba Lan), Philips Hue (Phần Lan),...

## 1.2. Nghiên cứu các tính năng nổi bật trong nhà thông minh hiện nay

- Tiết kiệm năng lượng.
- Kiểm soát hệ thống chiếu sáng thông minh.
- Kiểm soát môi trường, nhiệt độ, độ ẩm.
- Hệ thống điều khiển rèm cửa tự động.
- Điều khiển điều hòa thông minh.

- Điều khiển nhà thông minh thông qua giọng nói.
- Đảm bảo an ninh cho toàn bộ ngôi nhà.



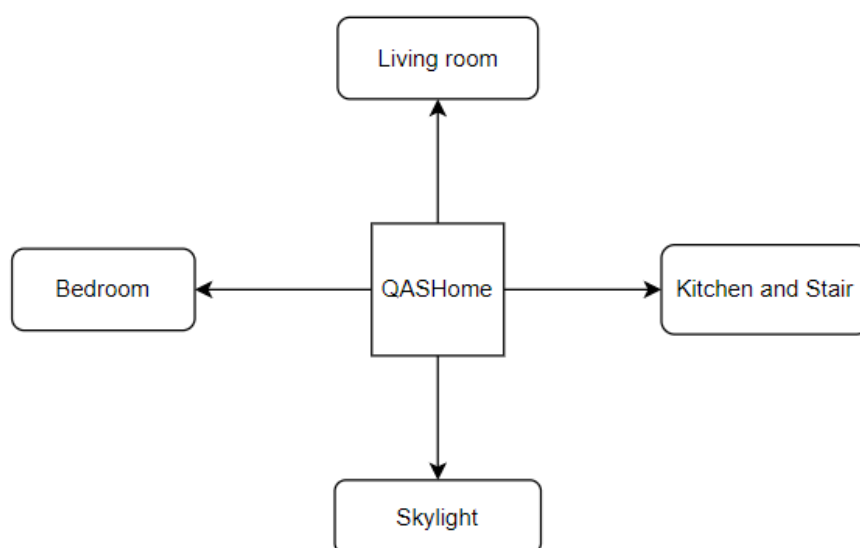
## CHƯƠNG 2. THIẾT KẾ MÔ HÌNH CHO ĐỀ TÀI

### 2.1. Lựa chọn hướng thiết kế cho đề tài

Hiện nay, ở Việt Nam, giải pháp xây dựng nhà thông minh với hệ thống điều khiển và giám sát thông qua internet vẫn phổ biến và phát triển hơn cả vì nó phù hợp với khả năng công nghệ và điều kiện kinh tế hiện có.

Nhóm chúng em nhận thấy đề tài mô hình nhà thông minh phù hợp với chuyên ngành Cơ Điện Tử cũng như dòng chảy công nghệ cao, vậy nên ở đề tài này, nhóm em xin chọn thiết kế ngôi nhà thông minh theo giải pháp sử dụng hệ thống điều khiển và giám sát các thiết bị trong ngôi nhà thông qua mạng internet, mà cụ thể là mạng wifi trên một mô hình nhà thông minh quy mô 600cm x 400cm (2 tầng) với các chức năng cơ bản như: cửa đóng/mở, giám sát và cảnh báo cháy, cảnh báo khí gas rò rỉ, đèn và quạt bật tự động theo người sử dụng và nhiệt độ môi trường.

### 2.2. Tính năng và yêu cầu đặt ra cho mô hình nhà thông minh của đề tài



*Hình 2. 1. Mô hình chung của nhà thông minh*

Mô hình nhà thông minh được nhóm đề xuất các tính năng như sau:

- Hệ thống phòng khách thực hiện các chức năng: Quản lý sử dụng đèn, quạt, đóng mở cửa, quản lý nhiệt độ, độ ẩm trong phòng. Việc quản lý được thực hiện tự động và thông qua app điện thoại trong thời gian thực nên yêu cầu thời gian trễ không quá lớn, xử lý chính xác.
- Hệ thống phòng bếp và cầu thang thực hiện các chức năng: Quản lý đèn cầu thang thông qua cảm biến, kiểm soát nhiệt độ, khí gas trong phòng bếp và cảnh báo nếu có dấu hiệu cháy nổ, quản lý sử dụng đèn, quạt. Việc quản lý được thực hiện tự động, thông qua app và cảnh báo nếu có tới tin nhắn SMS của người sử dụng nên yêu cầu thời gian trễ không quá lớn, tốc độ xử lý nhanh.



- c) Hệ thống các phòng ngủ thực hiện các chức năng: Quản lý sử dụng đèn, quạt, quản lý nhiệt độ, độ ẩm. Việc quản lý được thực hiện tự động và thông qua app điện thoại trong thời gian thực.
- d) Hệ thống khu vực giếng trời: Quản lý tình trạng mưa thời điểm hiện tại để đưa ra cảnh báo cũng như liên quan tới việc đóng, mở cửa giếng trời.

## **2.3. Các thành phần cơ bản trong nhà thông minh**

### **2.3.1. Hệ thống chiếu sáng**

Bao gồm hệ thống các đèn, có chức năng chiếu sáng cho các phòng theo yêu cầu của người sử dụng. Các đèn được bật tắt tùy thuộc vào yêu cầu của người sử dụng, tuy nhiên cũng có thể được bật tắt tự động nếu có sự xuất hiện của con người trong khu vực đặt cảm biến.

### **2.3.2. Hệ thống kiểm soát ra vào**

Bao gồm một camera, có chức năng nhận diện khuôn mặt của thành viên trong nhà từ đó có thể thực hiện điều khiển đóng mở hệ thống cửa ra vào.

### **2.3.3. Hệ thống cảnh báo rò rỉ khí gas**

Bao gồm cảm biến đo nồng độ khí gas hiện tại trong phòng bếp và nếu nồng độ khí gas vượt quá ngưỡng cảnh báo đặt trước thì sẽ gửi tin nhắn SMS về cho người sử dụng cảnh báo nồng độ khí gas đã vượt quá nồng độ cho phép, cần có biện pháp xử lý để tránh xảy ra cháy nổ.

### **2.3.4. Hệ thống kiểm soát môi trường**

Bao gồm cảm biến đo nhiệt độ, độ ẩm trong các phòng và gửi thông tin về máy điện thoại của người sử dụng. Các thông tin đó sẽ giúp chủ nhà có thể nắm được tình trạng hiện tại của căn phòng và có quyết định thực hiện bật tắt các thiết bị ngoại vi khác như quạt...

### **2.3.5. Hệ thống giếng trời**

Hiện tại đã có một số mô hình giếng trời tự động được lưu hành trên thị trường, tuy nhiên, những mô hình này đều có chung một đặc điểm là việc chiếm dụng diện tích sử dụng tương đối lớn. Diện tích này phụ thuộc vào kích thước của miệng giếng trời nên trong một số mô hình với diện tích giếng trời tương đương với diện tích phần sân thượng toà nhà sẽ rất khó để triển khai lắp đặt.

Nhóm nghiên cứu đề xuất thiết kế mô hình giếng trời tự động có thể tiết kiệm diện tích lắp đặt cũng như vận hành nhằm tối ưu hiệu năng sử dụng cho khu vực sân thượng toà nhà. Giờ đây phần diện tích còn lại có thể được sử dụng làm sân phơi, đặt bàn uống trà, thư giãn cho chủ căn hộ.



*Hình 2. 2. Mô hình mái và ray 1*



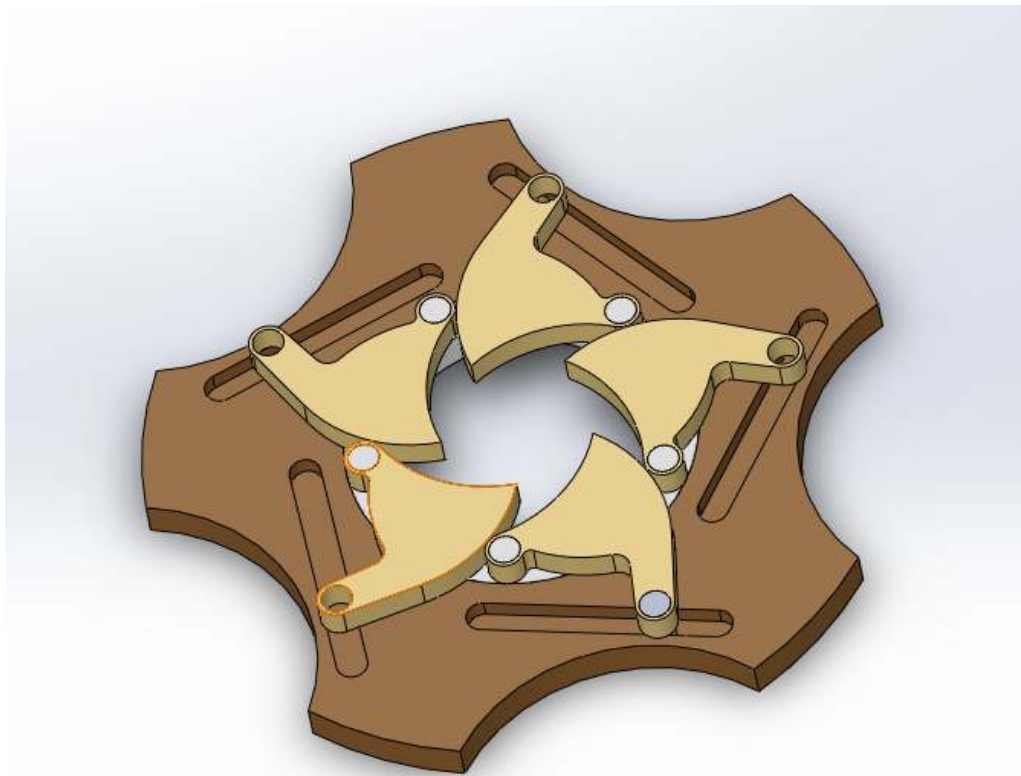
*Hình 2. 3. Mô hình má và ray 2*

#### **2.3.5.1. Yêu cầu mô hình:**

- Có khả năng đóng/ mở gọn gàng, tiết kiệm diện tích.
- Có tính thẩm mỹ.
- Người dùng có thể theo dõi và cài đặt tốc độ đóng mở qua ứng dụng điện thoại.

#### **2.3.5.2. Các mô hình đã thiết kế**

**Mô hình 1:** Mô hình thiết kế ban đầu



*Hình 2. 4. Mô hình thiết kế ban đầu*

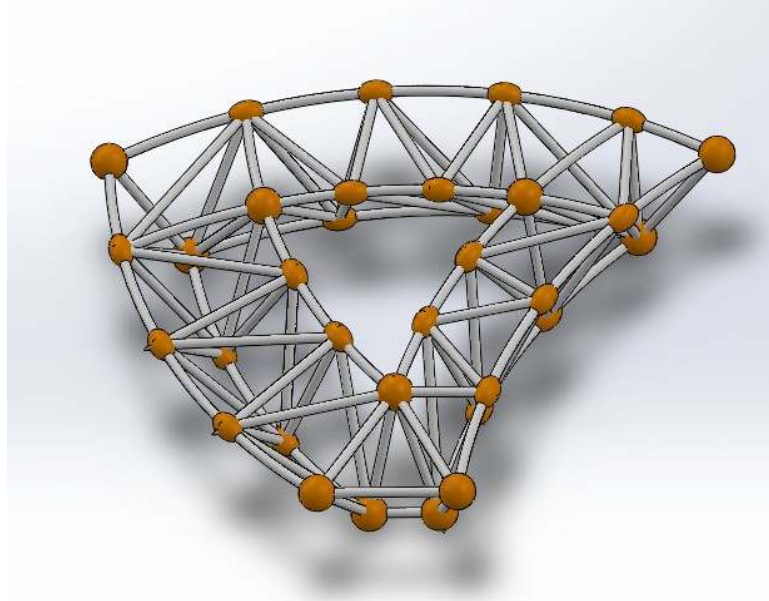
#### **\*Ưu điểm:**

- Diện tích hoạt động giới hạn bởi đĩa lớn bên ngoài.
- Thiết kế đối xứng, có tính thẩm mỹ.

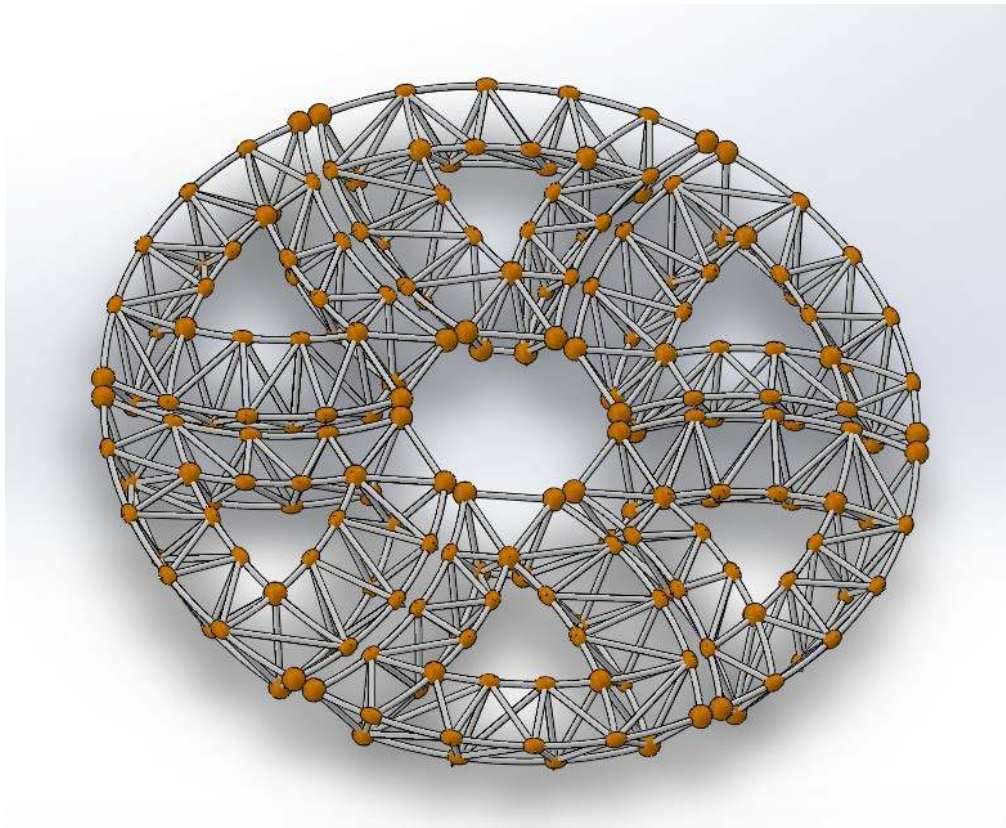
**\*Nhược điểm:**

- Khó chế tạo ở kích thước lớn (1m x 1m) do hình dạng chi tiết và vật liệu dạng tấm, ít chi tiết.
- Việc lắp đặt bộ truyền động, động cơ sẽ làm tốn thêm không gian sử dụng.

**Mô hình 2:** Mô hình mái dạng khung



*Hình 2. 5. Một cánh của mái*



*Hình 2. 6. Mô hình mái đầy đủ*

**\*Ưu điểm:**

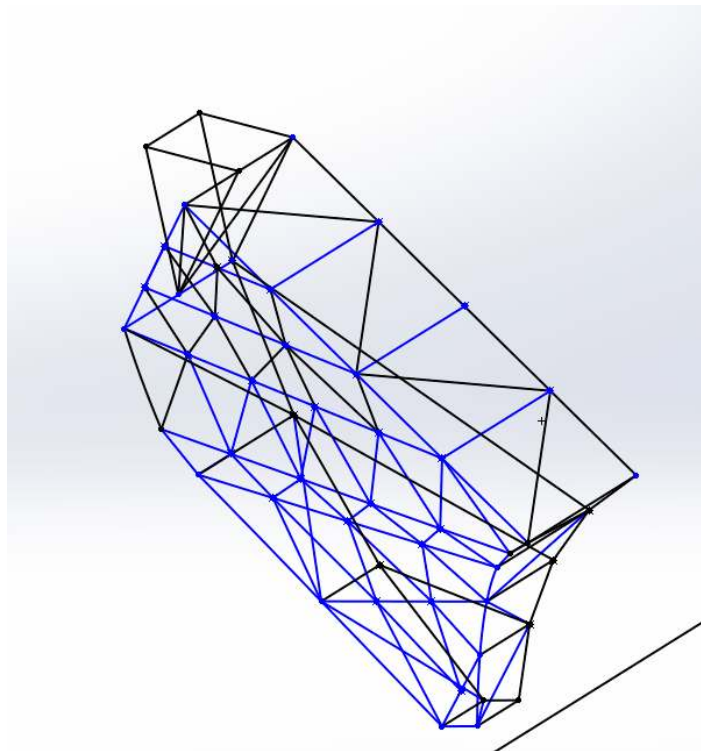
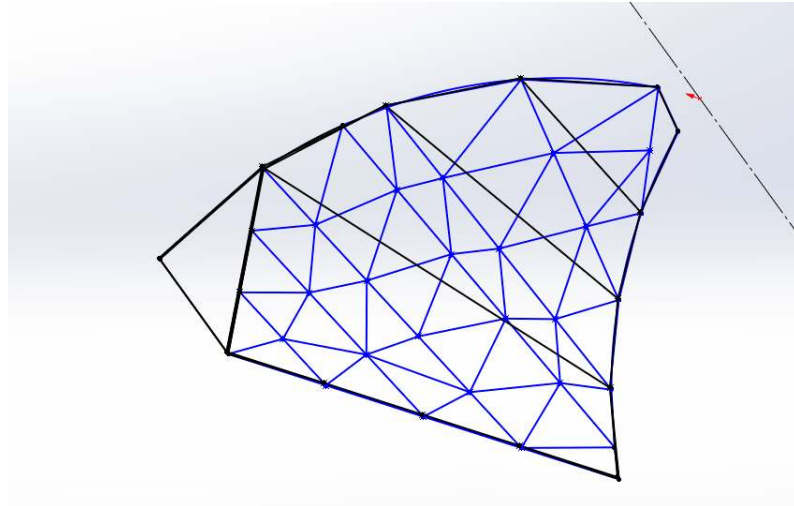
- Có thể triển khai ở kích thước lớn, dễ dàng lắp đặt.

- Thiết kế đối xứng, có tính thẩm mỹ.

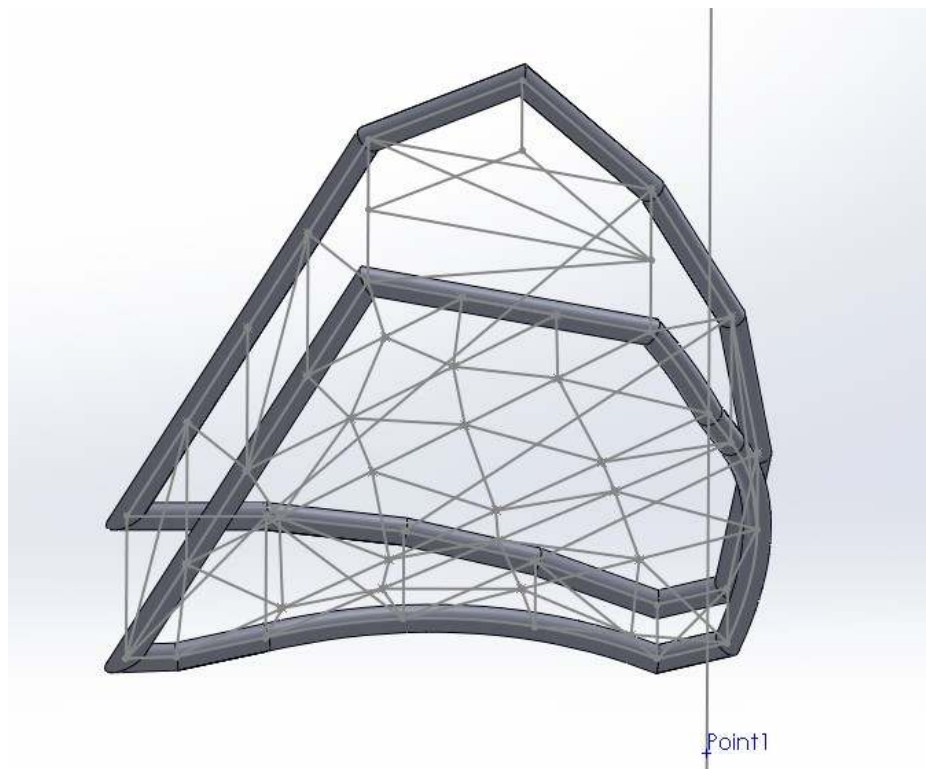
**\*Nhược điểm:**

- Hạn chế trong việc thiết kế hệ thống truyền động, cần lắp thêm ray, gia tăng diện tích hoạt động.

**Thiết kế mô hình dạng khung khác:**







*Hình 2. 7. Thiết kế mô hình khung dạng khác*

### **Mô hình 3: Mô hình mái che xếp sát thành giếng**



*Hình 2. 8. Mở giếng trời, mái thu*



*Hình 2. 9. Đóng giếng trời, mái xếp*

#### **\*Ưu điểm:**

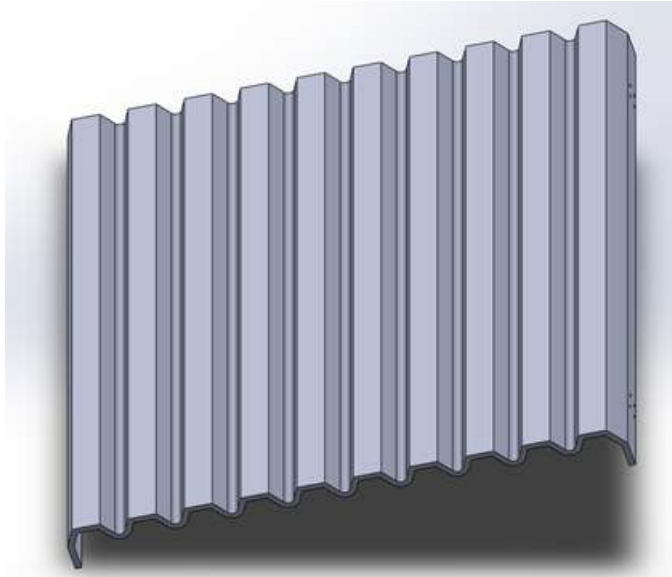
- Dễ dàng lắp đặt và triển khai ở kích thước lớn.
- Tùy vào công suất hoạt động có thể chọn các bộ truyền khí, thủy lực,... khác nhau để thay thế cho động cơ.

#### **\*Nhược điểm:**

- Tính thẩm mỹ chưa cao.

#### **\*Cấu tạo của mô hình:**

##### **a. Mái che giếng trời**



*Hình 2. 10. Mái che giếng trời*

Mái che được thiết kế để che được hết phần diện tích giếng trời, đồng thời những rãnh thoát nước sẽ tránh hiện tượng nước chảy ở rìa của mái vào bên trong giếng cũng như phần động cơ điện.

**b. Bộ phận nâng mái và động cơ**



*Hình 2. 11. Bộ phận nâng mái*

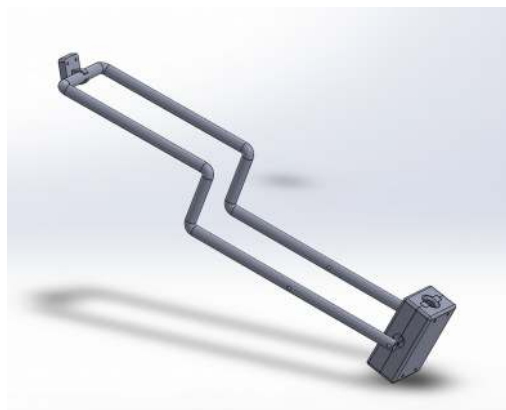


*Hình 2. 12. Động cơ*

Hai chân đế lắp động cơ được thiết kế dạng hộp với động cơ được gắn vào đế thông qua một bản lề.



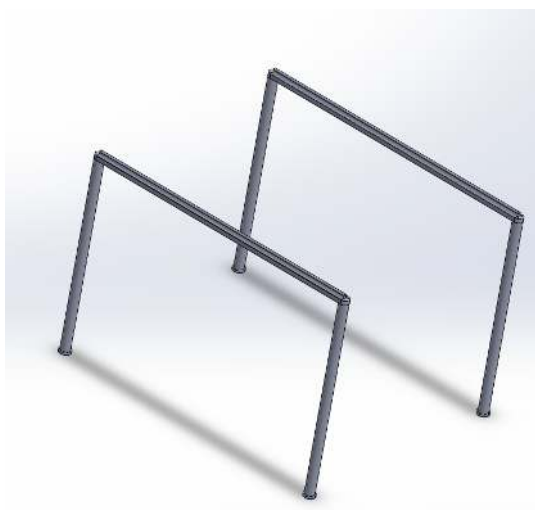
Hình 2. 13. Hộp gắn ren



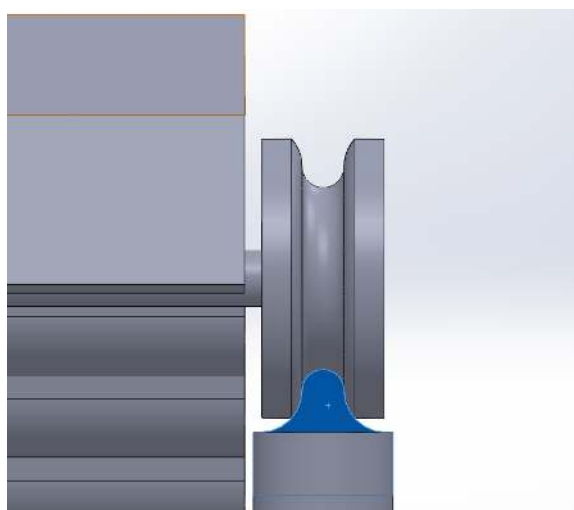
Hình 2. 14. Cụm thanh quay và hộp gắn ren

Thanh quay được gắn với hộp đai ốc, khi hộp này chuyển động dọc theo trục vít me, đầu thanh quay gắn với cửa được nâng lên, đẩy mái vào vị trí che giếng trời.

**c. Bộ phận ray dẫn và bánh ray**



Hình 2. 15. Thanh ray



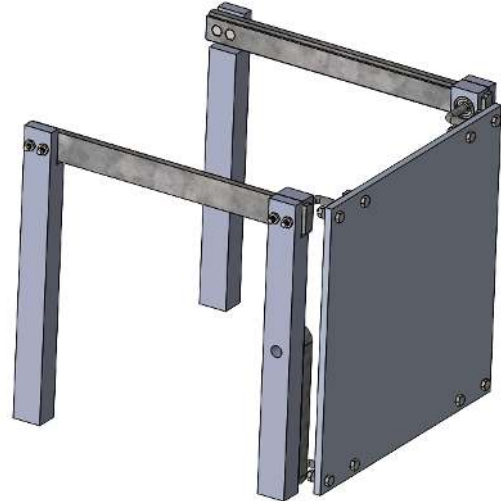
Hình 2. 16. Bánh ray ăn khớp với thanh ray

Hai ray dẫn được đặt trên hai trụ đỡ để dẫn hướng cho mái che. Mái che được gắn thêm 2 bánh xe ăn khớp với ray.

**Mô hình 4:** Mô hình mái trời cải tiến



Hình 2. 17. Đóng giếng trời



Hình 2. 18. Mở giếng trời

**\*Ưu điểm:**

- Sử dụng bộ truyền động được lắp đặt bên ngoài mô hình, có khả năng tùy biến cao.
- Sử dụng ray trượt, con trượt cho chuyển động mượt mà hơn mô hình 3

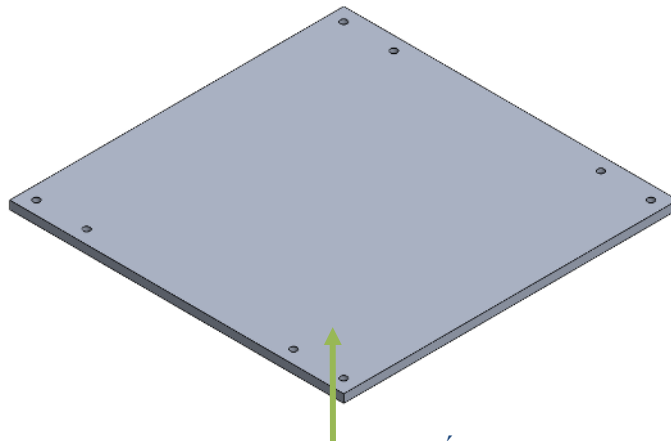
**\*Nhược điểm:**

- Tính thẩm mỹ chưa cao.

So sánh những ưu điểm, nhược điểm của các mô hình kể trên cùng với những tiêu chí đã đề ra từ ban đầu, nhóm nghiên cứu quyết định triển khai thiết kế theo mô hình 4 cho module cảnh báo mưa và giếng trời.

**\*Cấu tạo giếng trời:**

**a. Mái che giếng trời**



Hình 2. 19. Mái che giếng trời

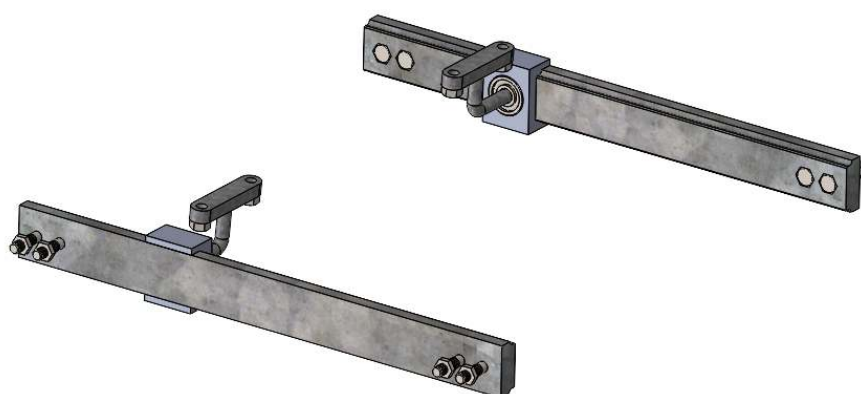
**b. Bộ phận nâng mái**





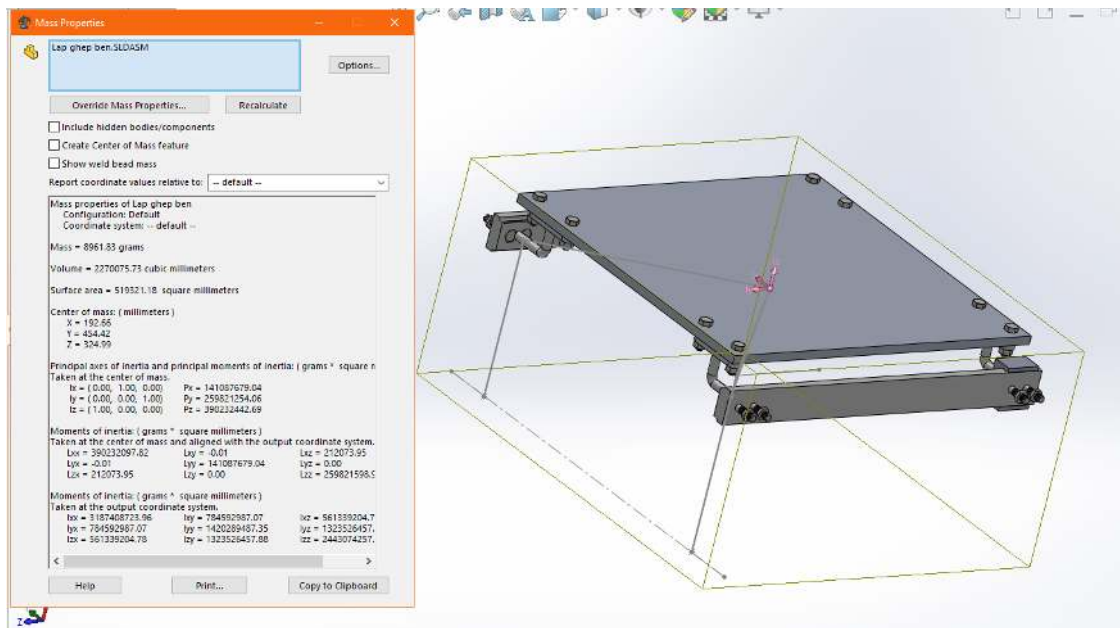
*Hình 2. 20. Bộ phận nâng mái*

**c. Bộ phận dẫn hướng**



*Hình 2. 21. Bộ phận dẫn hướng*

**\*Lựa chọn động cơ cho mô hình:**



Xét trọng lượng khung ở trọng tâm như trên. Ta dời lực về trục trục rồi đơn giản hóa như hình 2.22 . Ở đây chỉ xét momen làm cho trục quay nên ta có:  
(Momen thành phần làm quay trục)

Bộ truyền đai được thiết kế với tỷ số truyền 1:6, tỷ số momen giữa trục dẫn và trục bị dẫn là:

Trong quá trình chuyển động, để tăng tính an toàn cho hệ thống, ta sử dụng hệ số  $\eta$ , thu được momen động cơ cần cung cấp là:



## **CHƯƠNG 3.        NGHIÊN CỨU DỊCH VỤ FIREBASE ỨNG DỤNG TRONG TRUYỀN NHẬN DỮ LIỆU CỦA HỆ THỐNG IoT**

### **3.1. Giới thiệu về nền tảng Firebase**

Firebase là một platform do Google cung cấp, nhằm hỗ trợ việc tạo ra các web application, mobile application với chất lượng cao. Với việc sử dụng Firebase, các nhà phát triển có thể tập trung vào việc phát triển application mà không cần lo về việc sản phẩm của mình sẽ hoạt động và được quản lý thế nào ở phía Backend.

Khái niệm “Backend” được nhắc tới chỉ việc xử lý và lưu trữ dữ liệu trong Service, ở vùng “phía sau” mà người dùng không thể thấy được. Firebase là một trong những BaaS (Backend as a service).

#### **3.1.1. Các dịch vụ nổi bật của Firebase**

Để phát triển và kiểm thử các ứng dụng được thiết kế, Firebase đã tạo ra các công cụ:

- a) Realtime Database: Hỗ trợ đồng bộ hoá dữ liệu của người dùng kể cả khi không có kết nối mạng, tạo nên trải nghiệm xuyên suốt bất chấp tình trạng kết nối internet của người sử dụng. Dù Android, IOS, Web, C++, Unity... Realtime Database của Firebase vẫn hỗ trợ tốt.
- b) Crashlytics: Hệ thống theo dõi và lưu trữ thông tin lỗi của ứng dụng đang chạy trên máy người dùng. Nhờ Crashlytics, nhà phát triển có thể nắm bắt và xử lý kịp thời các lỗi chính của ứng dụng.
- c) Cloud Firestore: Lưu trữ và đồng bộ dữ liệu giữa người dùng và thiết bị sử dụng cơ sở dữ liệu noSQL được lưu trữ trên hạ tầng Cloud.
- d) Authentication: Bảo mật và đơn giản hoá quá trình quản lý người dùng. Firebase Auth cung cấp nhiều phương pháp để xác thực. Bao gồm email và mật khẩu, các nhà cung cấp bên thứ ba và sử dụng trực tiếp hệ thống tài khoản hiện tại của người dùng.
- e) Test Lab: Hỗ trợ chạy thử nghiệm tự động và tùy chỉnh cho ứng dụng của nhà phát triển trên cả thiết bị ảo lẫn vật lý do Google cung cấp.

Một số công cụ khác như: Performance Monitoring, Cloud Storage, Cloud Functions...

#### **3.1.2. Một số ứng dụng sử dụng nền tảng Firebase**

Một số ứng dụng khởi chạy trên nền tảng Firebase:

- a) Thời báo New York
- b) Alibaba
- c) Todoist
- d) eBay Motors

e) Le figaro

### 3.1.3. Ưu, nhược điểm khi sử dụng Firebase

#### a) Ưu điểm:

- Triển khai ứng dụng nhanh chóng: Firebase tiết kiệm rất nhiều thời gian quản lý và đồng bộ tất cả dữ liệu cho người dùng. Đó là nhờ người dùng không phải áp lực, quan tâm đến phần backend cùng các API tốt, hỗ trợ đa nền tảng.
- Bảo mật: Nhà phát triển có thể hoàn toàn yên tâm về độ bảo mật của Firebase nhờ nền tảng cloud, kết nối thông qua SSL, dùng JavaScript phân quyền người dùng cơ sở dữ liệu...
- Tính ổn định: Được viết dựa trên nền tảng cloud cung cấp bởi Google, các công cụ luôn đảm bảo độ ổn định tối đa. Bên cạnh đó, quá trình nâng cấp hay bảo trì Server cũng diễn ra nhanh hơn và đơn giản hơn.

#### b) Nhược điểm:

Cơ sở dữ liệu của Firebase được tổ chức theo kiểu trees, parent – children. Trong khi đó, người dùng SQL lại quen thuộc với kiểu Table truyền thống. Khi sử dụng Firebase, nhà phát triển sẽ mất một thời gian để làm quen trước khi sử dụng thành thạo.

### 3.2. Giới thiệu về dịch vụ Firebase Realtime Database

Firebase Realtime Database là một cơ sở dữ liệu NoSQL (cơ sở dữ liệu phi quan hệ) được lưu trữ và đồng bộ trên dịch vụ đám mây. Dữ liệu được đồng bộ trên tất cả các clients trong thời gian thực và vẫn khả thi khi ứng dụng offline.

Firebase Realtime Database là cơ sở dữ liệu lưu trữ trên mây. Dữ liệu được lưu trữ và đồng bộ hoá theo thời gian thực với mỗi client được kết nối. Khi nhà phát triển xây dựng ứng dụng đa nền tảng với iOS, Android và javascript SDK, tất cả các client chia sẻ một thể hiện Realtime Database và tự động tiếp nhận các thay đổi với dữ liệu mới nhất.

#### 3.2.1. Các khả năng chính của Firebase Realtime Database

##### a) Realtime:

Firebase Realtime Database sử dụng đồng bộ dữ liệu mới khi dữ liệu có sự thay đổi, mọi thiết bị được kết nối sẽ nhận được thay đổi trong vài mili giây.

##### b) Offline:

Khi người dùng ngoại tuyến, dữ liệu sẽ được lưu trên bộ nhớ cache của thiết bị và tự động đồng bộ lại khi bạn trực tuyến trở lại, tất cả đều là tự động.

##### c) Accessible from Client Devices:

Firebase Realtime Database có thể truy cập từ một thiết bị mobile hoặc trình duyệt web. Nó không cần một ứng dụng Server nào cả. Bảo mật và xác thực dữ liệu có thể thông qua các rule bảo mật của Firebase Realtime Database, các rule được thực thi khi dữ liệu được đọc hoặc ghi.

### 3.2.2. Lưu trữ dữ liệu trong Firebase Realtime Database

Firebase Realtime Database lưu trữ dữ liệu theo định dạng JSON. JSON là viết tắt của JavaScript Object Notion, là một dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được. Cú pháp của chuỗi JSON rất đơn giản, mỗi thông tin dữ liệu sẽ có 2 phần là “key” và “value”, điều này tương ứng trong CSDL là tên field và giá trị của nó ở một record nào đấy. Ví dụ khi định nghĩa một chuỗi JSON như sau:

```
{
  "username" : "kimoanh",
  "email" : "kimoan@gmail.com",
  "website" : "json.org",
  "title" : "Tìm hiểu về JSON"
}
```

Hình 3. 1. Ví dụ về một chuỗi JSON

### 3.2.3. Thực hiện các thao tác CRUD (Creat – Read – Update – Delete)

Để thực hiện bất kì phương thức nào trên cơ sở dữ liệu cho dù có thể đọc hoặc ghi đều cần phải có được các tham chiếu đến cơ sở dữ liệu. Để có thể tham chiếu đến nút trên cùng của cơ sở dữ liệu JSON, thực hiện:

```
private DatabaseReference mDatabase;

mDatabase = FirebaseDatabase.getInstance().getReference();
```

Hình 3. 2. Tham chiếu tới cơ sở dữ liệu JSON

#### a) Inserting Data:

Để thêm dữ liệu, ta sử dụng phương thức setValue() trên đường dẫn tham chiếu đến database. Nó sẽ tạo mới và cập nhật giá trị trên đường dẫn được cung cấp. Ví dụ ở dưới đã mã để thêm 1 nút được gọi là “copyright” trên cây json ở mức đỉnh:

```
DatabaseReference mRef = mDatabase.getReference("copyright");

mRef.setValue("@2016 androidhive. All rights Reserved");
```

Hình 3. 3. Minh họa với thao tác Insert data

Realtime database chấp nhận nhiều loại dữ liệu: String, Long, Double, Boolean, Map<String, Object>, List<Object> để lưu trữ dữ liệu. Ta cũng có thể sử dụng dữ liệu tùy biến của đối tượng để lưu trữ dữ liệu, điều này rất hữu dụng khi lưu trữ đối tượng vào database một cách trực tiếp.

Mọi user cần một ID duy nhất, bạn có thể tạo ra một bằng cách gọi phương thức `push()` để tạo ra một nút trống rỗng, với khóa duy nhất. Sau đó, được tham chiếu đến nút 'user' bằng phương thức `child()`. Cuối cùng sử dụng phương thức `setValue()` để lưu trữ dữ liệu người dùng.

#### b) Reading Data:

Để đọc dữ liệu, cần đính kèm phương thức `ValueEventListener()` để tham chiếu database. Sự kiện này sẽ được kích hoạt bất cứ khi nào có sự thay đổi trong dữ liệu trong thời gian thực. Trong `onDataChange()`, ta có thể thực hiện các phương thức mong muốn vào dữ liệu mới.

Dưới đây là sự kiện lắng nghe được kích hoạt bất cứ khi nào có sự thay đổi trong dữ liệu hồ sơ người dùng mà đã tạo ra trước đó.

```
mDatabase.child(userId).addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {

        User user = dataSnapshot.getValue(User.class);

        Log.d(TAG, "User name: " + user.getName() + ", email " + user.getEmail());
    }

    @Override
    public void onCancelled(DatabaseError error) {
        // Failed to read value
        Log.w(TAG, "Failed to read value.", error.toException());
    }
});
```

*Hình 3. 4. Minh họa với thao tác Reading data*

#### c) Updating Data:

Để cập nhật dữ liệu, ta có thể sử dụng cùng phương pháp `setValue()` để passing giá trị mới. Mặt khác cũng có thể sử dụng phương thức `updateChildren()` để passing đường dẫn để cập nhật dữ liệu mà không làm ảnh hưởng đến các nút con khác. Ví dụ khi ta cập nhật lại địa chỉ email của người dùng.

```
String newEmail = 'androidhive@gmail.com';

mDatabase.child(userId).child("email").setValue(newEmail);
```

*Hình 3. 5. Minh họa với thao tác Updating data*

#### d) Deleting Data:

Để xóa dữ liệu, ta có thể gọi phương thức `removeValue()` trong tham chiếu database. Ta cũng có thể pass qua null để gọi phương thức `setValue()`, nó giống như phương thức xóa.

### 3.3. Ứng dụng của Firebase Realtime Database vào đề tài nhà thông minh



Từ những phân tích trên về chức năng và phương thức quản lý dữ liệu thời gian thực trên Firebase Realtime Database, nhóm chúng em quyết định sử dụng dịch vụ Firebase Realtime Database để lưu trữ và quản lý dữ liệu cho dự án.

Vì cấu trúc quản lý dữ liệu của Firebase Realtime Database có dạng cấu trúc cây, tại mỗi nút của cây có dạng key – value (vì Firebase Realtime Database hỗ trợ rất đa dạng các kiểu cho value, tuy nhiên để thống nhất giữa các thành viên khi thực hiện đồ án, nhóm chúng em quyết định sử dụng kiểu String cho tất cả các value) nên nhóm sẽ thực hiện quản lý mỗi chức năng (từng phòng, quản lý điều kiện sống) thành một parent, trong mỗi parent có các children riêng biệt tùy thuộc vào từng chức năng. Cụ thể, nhóm thực hiện tạo 10 parent bao gồm:



*Hình 3. 6. Cấu trúc quản lý dữ liệu của dự án*

10 parent trên bao gồm: livingroom (quản lý thông tin sử dụng của phòng khách), kitchen (quản lý thông tin sử dụng của phòng bếp), bedroom1 (quản lý thông tin sử dụng của phòng ngủ 1), bedroom2 (quản lý thông tin sử dụng của phòng ngủ 2), bathroom (quản lý thông tin sử dụng của phòng vệ sinh), skylight (quản lý thông tin sử dụng của khu vực giếng trời), lifecondition (quản lý dữ liệu điều kiện môi trường khu vực), login (quản lý thông tin đăng nhập username và password của người sử dụng), register (quản lý thông tin đăng ký tài khoản mới), forgot\_password (quản lý thông tin lấy lại thông tin tài khoản người dùng).

Mỗi parent trên sẽ có các children nhỏ hơn quản lý thông tin cụ thể, bao gồm:

**a) Parent livingroom:**

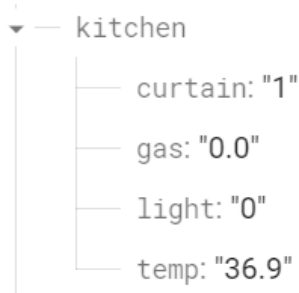




*Hình 3. 7. Quản lý dữ liệu của phòng khách*

Trong parent livingroom, thực hiện quản lý các thông tin: door (thông tin người dùng đóng/mở cửa ra vào tương ứng với hai giá trị “0” và “1”); fan (thông tin người dùng tắt/bật quạt tương ứng với hai giá trị “0” và “1”); light (thông tin người dùng tắt/bật đèn ứng với hai giá trị “0” và “1”); temp (thông tin điều kiện nhiệt độ trong khu vực phòng khách được đọc từ cảm biến trong phòng); humid (thông tin điều kiện độ ẩm trong khu vực phòng khách được đọc từ cảm biến trong phòng).

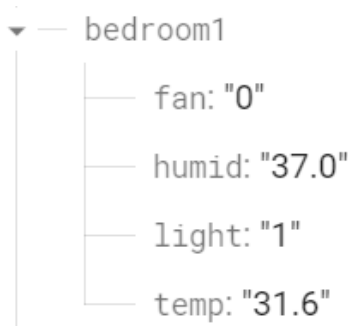
**b) Parent kitchen:**



*Hình 3. 8. Quản lý dữ liệu của phòng bếp*

Trong parent kitchen, thực hiện quản lý các thông tin: light (thông tin người dùng tắt/bật đèn ứng với hai giá trị “0” và “1”); curtain (thông tin người dùng đóng/mở cửa sổ thoán ứng với hai giá trị “0” và “1”); temp (thông tin điều kiện nhiệt độ trong khu vực phòng bếp được đọc từ cảm biến trong phòng); gas (thông tin điều kiện nồng độ khí gas được đọc từ cảm biến được đặt trong phòng).

**c) Parent bedroom1:**

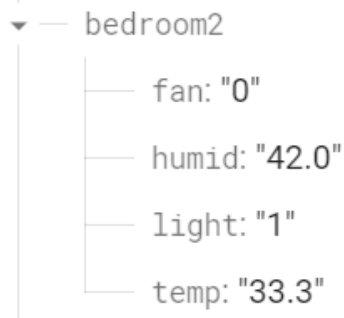


*Hình 3. 9. Quản lý dữ liệu của phòng ngủ 1*

Trong parent bedroom1, thực hiện quản lý các thông tin: light (thông tin người dùng tắt/bật đèn ứng với hai giá trị “0” và “1”); (thông tin người dùng tắt/bật quạt tương ứng với hai giá trị “0” và “1”); temp (thông tin điều kiện nhiệt độ trong

khu vực phòng khách được đọc từ cảm biến trong phòng); humid (thông tin điều kiện độ ẩm trong khu vực phòng khách được đọc từ cảm biến trong phòng).

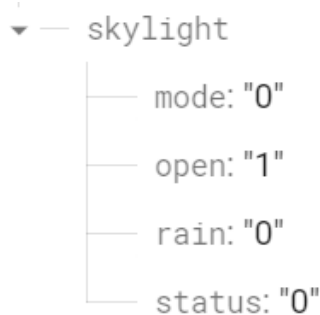
**d) Parent bedroom2:**



*Hình 3. 10. Quản lý dữ liệu của phòng ngủ 2*

Parent bedroom2 tương tự với parent bedroom1, thực hiện quản lý các thông tin: light (thông tin người dùng tắt/bật đèn ứng với hai giá trị “0” và “1”); (thông tin người dùng tắt/bật quạt tương ứng với hai giá trị “0” và “1”); temp (thông tin điều kiện nhiệt độ trong khu vực phòng khách được đọc từ cảm biến trong phòng); humid (thông tin điều kiện độ ẩm trong khu vực phòng khách được đọc từ cảm biến trong phòng).

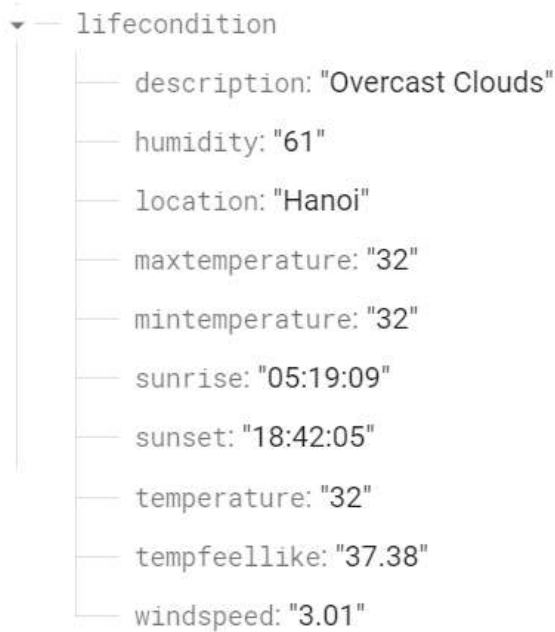
**e) Parent skylight:**



*Hình 3. 11. Quản lý dữ liệu khu vực giếng trời*

Parent skylight thực hiện quản lý các thông tin: mode (thông tin người dùng lựa chọn 1 trong 2 chế độ đóng mở giếng trời là automatic và manual tương ứng là “0” và “1”); open (thông tin người dùng sau khi chọn mở bằng chế độ manual sẽ thực hiện đóng/mở giếng trời tương ứng là “0” và “1”); rain (thông tin về tình trạng mưa hiện tại, được ghi nhận từ cảm biến lắp đặt ở khu vực giếng trời).

**f) Parent lifecondition:**



*Hình 3. 12. Quản lý dữ liệu về điều kiện khu vực*

Parent lifecondition thực hiện quản lý thông tin điều kiện trong khu vực, được lấy từ OpenWeatherMap của OpenWeather Ltd. OpenWeatherMap cho phép sử dụng API để truy xuất thông tin thời tiết tại khu vực người dùng yêu cầu tại thời điểm hiện tại hoặc trong một tuần tiếp theo. Thông tin được lưu dưới định dạng JSON được hỗ trợ đọc ghi thuận tiện bằng ngôn ngữ python.

```

1  {
2      "coord": {
3          "lon": 105.8412,
4          "lat": 21.0245
5      },
6      "weather": [
7          {
8              "id": 804,
9              "main": "Clouds",
10             "description": "overcast clouds",
11             "icon": "04d"
12         }
13     ],
14     "base": "stations",
15     "main": {
16         "temp": 32,
17         "feels_like": 34.61,
18         "temp_min": 32,
19         "temp_max": 32,
20         "pressure": 1002,
21         "humidity": 51,
22         "sea_level": 1002,
23         "grnd_level": 1001
24     },
25     "visibility": 10000,
26     "wind": {
27         "speed": 1.15,
28         "deg": 205,
29         "gust": 3.11
30     },
31     "clouds": {
32         "all": 100
33     },
34     "dt": 1652166809,
35     "sys": {
36         "type": 1,
37         "id": 9308,
38         "country": "VN",
39         "sunrise": 1652134885,
40         "sunset": 1652181884

```

*Hình 3. 13. File JSON data*

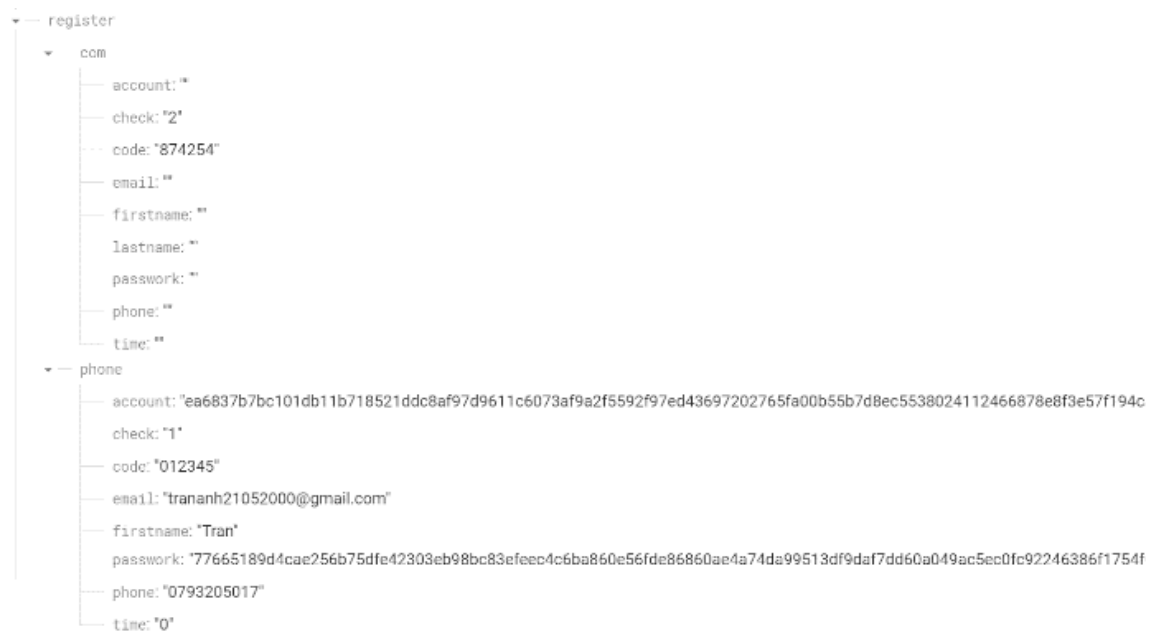
Các điều kiện về thời tiết được nhóm thu thập và tạo lập thành thông tin cho người sử dụng bao gồm: description (mô tả thời tiết chung của khu vực); humidity (độ ẩm); location (vị trí của khu vực); maxtemperature và mintemperature (nhiệt độ cao nhất và thấp nhất trong ngày); sunrise và sunset (thời gian mặt trời mọc và lặn trong khu vực); temperature (nhiệt độ khu vực); tempfeellike (nhiệt độ cảm thấy thực tế) và windspeed (tốc độ gió).

#### g) Parent login:



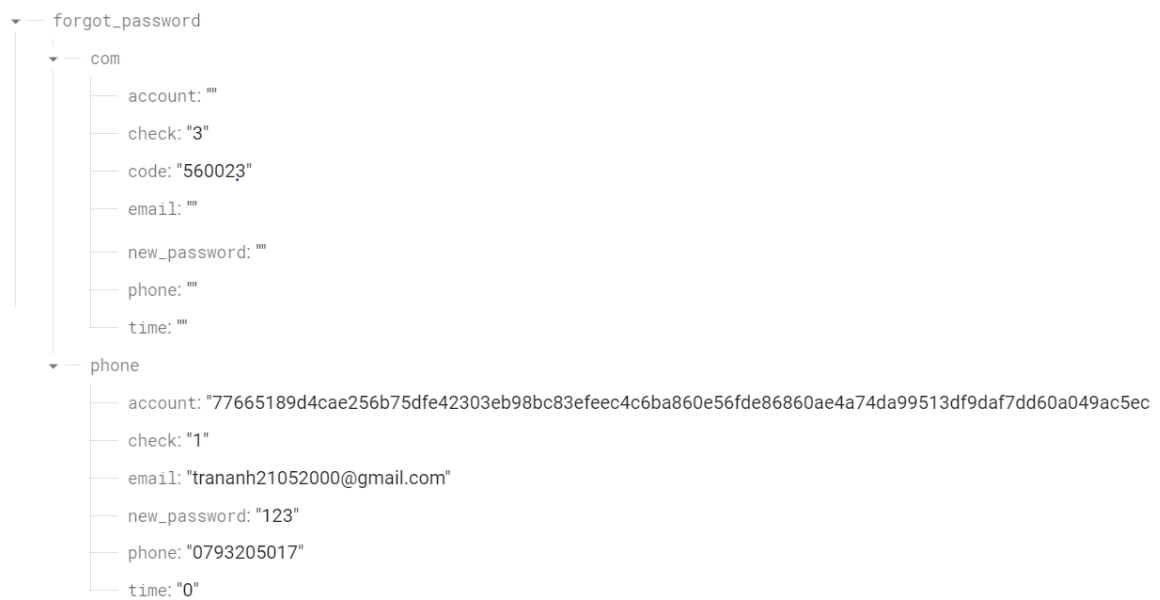
Hình 3. 14. Quản lý dữ liệu về thông tin đăng nhập người dùng

#### h) Parent register:



Hình 3. 15. Quản lý dữ liệu đăng ký người dùng

#### i) Parent forgot\_password:



*Hình 3. 16. Quản lý dữ liệu khi muốn lấy lại thông tin đã quên*

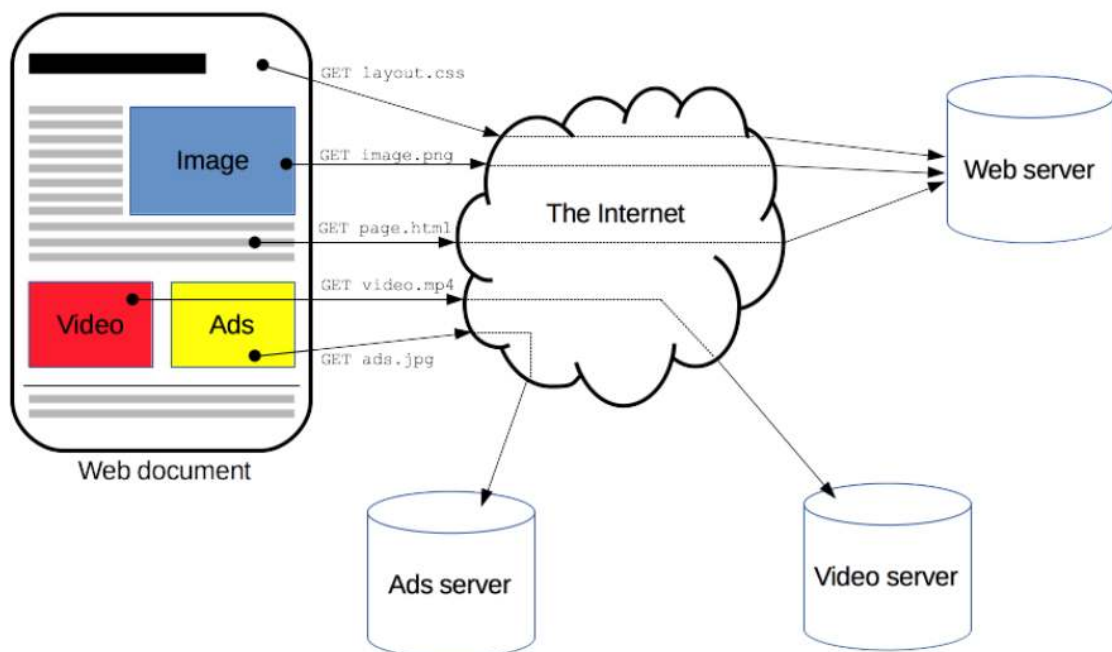
## CHƯƠNG 4.           NGHIÊN CỨU GIAO THỨC TRUYỀN NHẬN DỮ LIỆU HTTP VÀ MQTT

### 4.1. Nghiên cứu sử dụng giao thức HTTP truyền nhận dữ liệu thông tin giữa app và bộ điều khiển chức năng nhà thông minh

#### 4.1.1. Giới thiệu giao thức HTTP

HTTP là viết tắt của Hyper Text Transfer Protocol, tức là Giao thức truyền tải siêu văn bản được sử dụng trong www. HTTP là một giao thức cho phép tìm, nạp tài nguyên chẳng hạn như HTML doc.

HTTP là nền tảng của bất kì sự trao đổi dữ liệu nào trên WEB và cũng là giao thức giữa client (thường là các trình duyệt hoặc bất kì một loại thiết bị nào, chương trình nào) và server (thường là các máy tính trên đám mây).



Hình 4. 1. Giao thức truyền tải HTTP

Được thiết kế lần đầu tiên từ những năm 90, HTTP là một giao thức có thể mở rộng vốn đã phát triển dần theo thời gian. Một giao thức lớp ứng dụng được gửi thông qua nền tảng TCP/IP, hay qua một kết nối TCP được mã hoá TLS. Mặc dù về mặt lý thuyết, bất kỳ giao thức truyền tải đáng tin nào cũng có thể được sử dụng.

Nhờ vào khả năng mở rộng của HTTP, nó được sử dụng để không chỉ tìm nạp các tài liệu siêu văn bản, mà còn cả hình ảnh và video hoặc để đăng tải nội dung lên server.

#### 4.1.2. Các phương thức sử dụng trong giao thức HTTP

HTTP Request là thông tin được gửi từ client lên server, để yêu cầu server tìm hoặc xử lý một số thông tin, dữ liệu mà client mong muốn. HTTP Request có thể là một file text dưới dạng XML hoặc Json mà cả server và client đều có thể hiểu được.

Các phương thức của HTTP Request bao gồm:

#### **a) Phương thức GET**

Get là phương thức được Client lấy lại thông tin dữ liệu từ Server thông qua đường dẫn URL nằm trên thanh địa chỉ của Browser. Server sẽ nhận đường dẫn đó và phân tích sau đó trả về kết quả cho Client. Hơn nữa, đó là một phương thức được sử dụng phổ biến mà không cần có Request Body.

Một số đặc điểm chính của phương thức Get:

- Giới hạn độ dài của các giá trị là 255
- Chỉ hỗ trợ các dữ liệu có kiểu String
- Có thể lưu bộ nhớ vào Cache
- Các tham số truyền vào được lưu trữ trong lịch sử trình duyệt
- Có thể được đánh dấu và xem lại do được lưu trong lịch sử trình duyệt

#### **b) Phương thức POST**

Phương thức Post là phương thức gửi dữ liệu đến Server giúp Client có thể thêm mới dữ liệu hoặc cập nhật dữ liệu đã có vào database.

Một số đặc điểm chính của phương thức Post là:

- Dữ liệu cần thêm hoặc cập nhật không được hiển thị trong URL của trình duyệt
- Dữ liệu không được lưu trong lịch sử trình duyệt
- Không có hạn chế về độ dài của dữ liệu
- Hỗ trợ nhiều kiểu dữ liệu như: String, Binary, Integers...

#### **c) Phương thức PUT**

Cách thức hoạt động của phương thức Put tương tự như phương thức Post tuy nhiên nó chỉ được sử dụng để cập nhật dữ liệu đã có trong Database. Khi sử dụng nó, Client phải sử dụng toàn bộ dữ liệu của một đối tượng.

#### **d) Phương thức PATCH**

Tương tự như phương thức Post và phương thức Put, phương thức Patch được sử dụng khi phải cập nhật một phần dữ liệu của đối tượng.

#### **e) Phương thức DELETE**

Khi sử dụng phương thức Delete, sẽ thực hiện xóa các dữ liệu của Server về tài nguyên thông qua URL. Cũng giống như Get, phương thức này không có Body Request.

#### **f) Phương thức HEAD**

Phương thức Head có nhiều điểm gần giống với phương thức Get, tuy nhiên phương thức Head không có response body. Hay nói cách khác, nếu sử dụng phương thức Get tới đường dẫn (Books) thì sẽ trả về danh sách các sản phẩm, còn khi sử dụng phương thức Head tới đường dẫn (Books) thì sẽ nhận được danh sách các sản phẩm.

#### 4.1.3. Ứng dụng giao thức HTTP trong đề tài

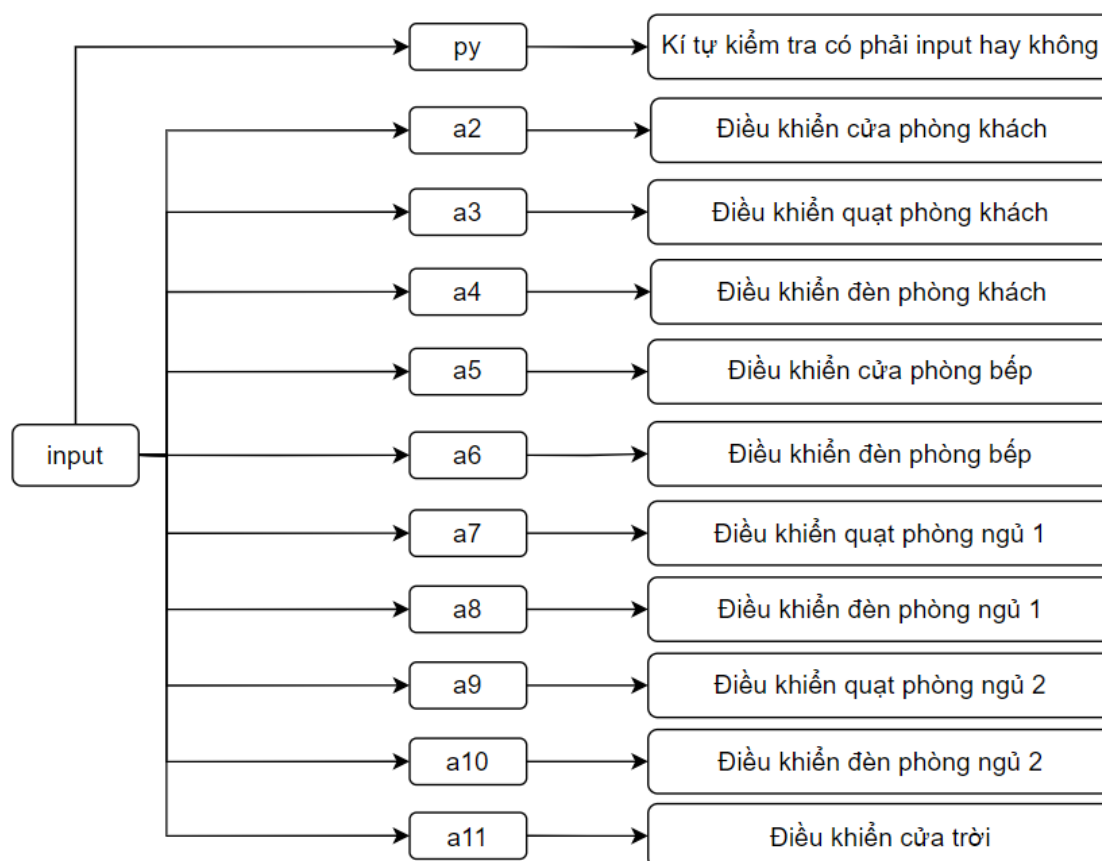
Trong phạm vi đề tài, chỉ sử dụng 2 phương thức là Get và Post cho việc lấy và nhận dữ liệu giữa Server và vi điều khiển Arduino. Cụ thể, phương thức Post dùng khi vi điều khiển Arduino gửi đến Server thông tin thu thập được từ các cảm biến trong nhà và phương thức Get dùng khi vi điều khiển nhận thông tin từ Server tín hiệu điều khiển các thiết bị trong nhà.

Nhóm thực hiện quy ước quá trình trao đổi thông tin giữa Server và Client là quá trình trao đổi các chuỗi String được quy ước sẵn, thực hiện lưu trữ thông tin dữ liệu phục vụ cho quá trình trao đổi dữ liệu.

##### a) Dữ liệu với phương thức Get

Phương thức Get được sử dụng cho quá trình vi điều khiển nhận thông tin từ Server các tín hiệu điều khiển các thiết bị trong nhà. Dữ liệu được lưu trữ là một chuỗi String gồm 12 ký tự như sau:

$input = "pya_2a_3a_4a_5a_6a_7a_8a_9a_{10}"$



Hình 4. 2. Cấu trúc tín hiệu input

Trong đó 2 ký tự đầu tiên là  $py$  là ký tự được nhóm quy ước cho tín hiệu sử dụng với phương thức Get.

Các ký tự còn lại được giải thích như sau:

- Ký tự  $a_2$  được quy ước để thực hiện đóng/mở cửa khu vực phòng khách tương ứng với 2 giá trị dạng String là "0" và "1"



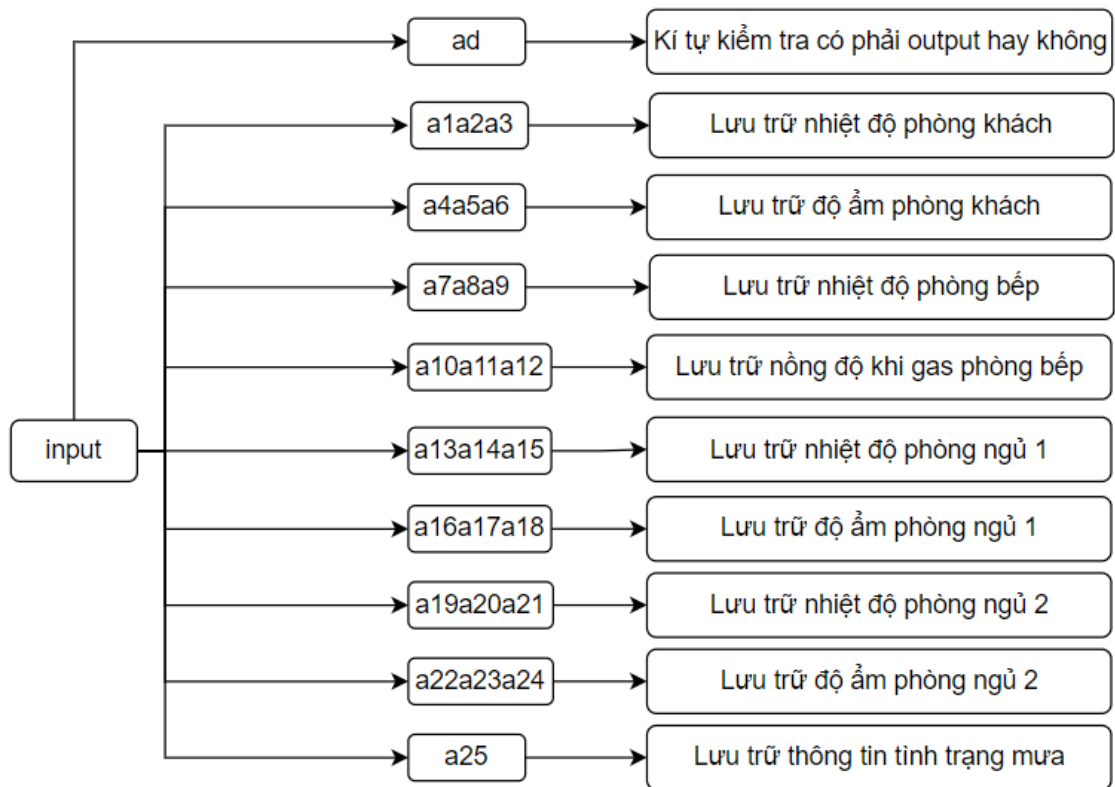
- Kí tự  $a_3$  được quy ước để thực hiện bật/tắt quạt khu vực phòng khách tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_4$  được quy ước để thực hiện bật/tắt đèn khu vực phòng khách tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_5$  được quy ước để thực hiện đóng/mở cửa khu vực phòng bếp tương ứng với 2 giá trị dạng String là "0" và "1"
- Kí tự  $a_6$  được quy ước để thực hiện bật/tắt đèn khu vực phòng bếp tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_7$  được quy ước để thực hiện bật/tắt quạt khu vực phòng ngủ thứ nhất tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_8$  được quy ước để thực hiện bật/tắt đèn khu vực phòng ngủ thứ nhất tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_9$  được quy ước để thực hiện bật/tắt quạt khu vực phòng ngủ thứ hai tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_{10}$  được quy ước để thực hiện bật/tắt đèn khu vực phòng ngủ thứ hai tương ứng với 2 giá trị dạng String là "1" và "0"
- Kí tự  $a_{11}$  được quy ước để thực hiện đóng mở cửa trời tương ứng với 2 giá trị dạng String là "1" và "0"

Tín hiệu String input được gửi liên tục nhờ phương thức Get, nhờ vậy mà vi điều khiển có thể thực hiện điều khiển các thiết bị theo mong muốn của người sử dụng trong thời gian thực.

#### **b) Dữ liệu với phương thức Post**

Phương thức Post được sử dụng trong quá trình gửi dữ liệu của các cảm biến sau khi được thu thập bởi vi điều khiển lên Server. Tương tự với dữ liệu của phương thức Get, dữ liệu của phương thức Post cũng là một chuỗi String với 25 kí tự thành từng cặp. Cụ thể:

$output = "ad a_1 a_2 a_3 a_4 a_5 a_6 a_7 a_8 a_9 a_{10} a_{11} a_{12} a_{13} a_{14} a_{15} a_{16} a_{17} a_{18} a_{19} a_{20} a_{21} a_{22} a_{23} a_{24} a_{25}"$



Hình 4. 3. Cấu trúc tín hiệu output

Trong đó 2 kí tự đầu là  $ad$  được quy ước tín hiệu là tín hiệu sử dụng với phương thức Post.

Các kí tự còn lại được giải thích như sau:

- 2 chuỗi String  $\overline{a_1a_2a_3}$  và  $\overline{a_4a_5a_6}$  dùng để lưu trữ thông tin về nhiệt độ và độ ẩm của khu vực phòng khách, là các chuỗi String 3 chữ số.
- 2 chuỗi String  $\overline{a_7a_8a_9}$  và  $\overline{a_{10}a_{11}a_{12}}$  dùng để lưu trữ thông tin về nhiệt độ và nồng độ khí gas của khu vực phòng bếp, là các chuỗi String 3 chữ số.
- 2 chuỗi String  $\overline{a_{13}a_{14}a_{15}}$  và  $\overline{a_{16}a_{17}a_{18}}$  dùng để lưu trữ thông tin về nhiệt độ và độ ẩm của khu vực phòng ngủ thứ nhất, là các chuỗi String 3 chữ số.
- 2 chuỗi String  $\overline{a_{19}a_{20}a_{21}}$  và  $\overline{a_{22}a_{23}a_{24}}$  dùng để lưu trữ thông tin về nhiệt độ và độ ẩm của khu vực phòng ngủ thứ hai, là các chuỗi String 3 chữ số.
- Chuỗi String  $a_{25}$  dùng để lưu trữ thông tin về tình trạng mưa hay không từ cảm biến mưa, tương ứng với 2 giá trị là “1” và “0”

Tín hiệu String output được cập nhật liên tục từ vi điều khiển, thông qua phương thức Post và gửi về Server trong thời gian thực. Vì vậy mà người dùng có thể nắm được thông tin về điều kiện môi trường trong nhà của mình.

#### 4.2. Nghiên cứu sử dụng giao thức MQTT truyền nhận dữ liệu thông tin giữa app android và Firebase Realtime Database

#### 4.2.1. Giới thiệu giao thức MQTT

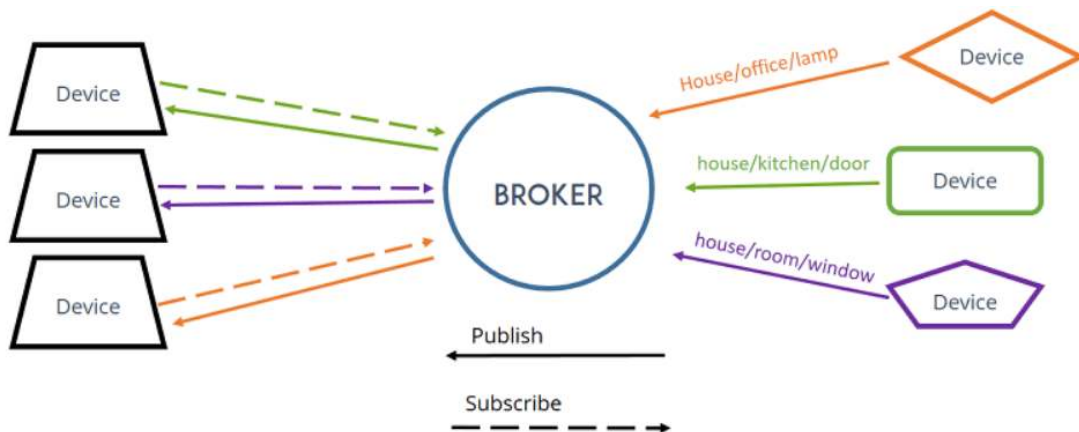
MQTT là viết tắt của cụm từ Message Queue Telemetry Transport, đây là một giao thức truyền thông điệp (Message) theo mô hình Publish – Subscriber (Cung cấp – Thuê bao), sử dụng băng thông thấp, độ tin cậy cao và có khả năng hoạt động trong điều kiện đường truyền không ổn định.

Kiến trúc mức cao (high – level) của MQTT gồm 2 phần chính là Broker (Máy chủ môi giới) và Clients. Các Broker tương đối “nhẹ” (khá ít các tác vụ xử lý) và được thiết kế có tính mở (tức là không đặc trưng cho riêng ứng dụng cụ thể nào cả).

Giao thức MQTT là một lựa chọn lý tưởng trong các môi trường như:

- Những nơi mà giá mạng viễn thông đắt đỏ, băng thông thấp hay thiếu tin cậy
- Khi chạy trên những thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ
- Giao thức MQTT sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M (Machine to Machine)

#### 4.2.2. Nguyên lý làm việc của giao thức MQTT



Hình 4. 4. Nguyên lý làm việc của giao thức MQTT

Broker hay máy chủ môi giới được coi như là trung tâm, nó là điểm giao của tất cả các kết nối đến từ các Client (bao gồm Publisher và Subscriber). Nhiệm vụ chính của Broker là nhận thông điệp (Message) từ các Publisher, xếp các Message theo hàng đợi rồi chuyển chúng tới một địa chỉ cụ thể. Ngoài ra, Broker còn có nhiệm vụ nhỏ hơn là đảm nhận một vài tính năng liên quan đến quá trình truyền như: bảo mật Message, lưu trữ Message, Logs...

Client là các thiết bị, ứng dụng được kết nối tới Broker để thực hiện truyền nhận dữ liệu. Các Client lại được chia thành 2 nhóm là Publisher và Subscriber. Một Client có thể thực hiện 1 trong 2 nhiệm vụ, hoặc thực hiện cả 2 nhiệm vụ như sau:

- Publisher là Client (thiết bị) gửi bản tin lên Broker
- Subscriber là Client (thiết bị) nhận bản tin mỗi khi có bản tin mới được gửi lên Broker

#### 4.2.3. Ưu nhược điểm của giao thức MQTT

##### a) Ưu điểm

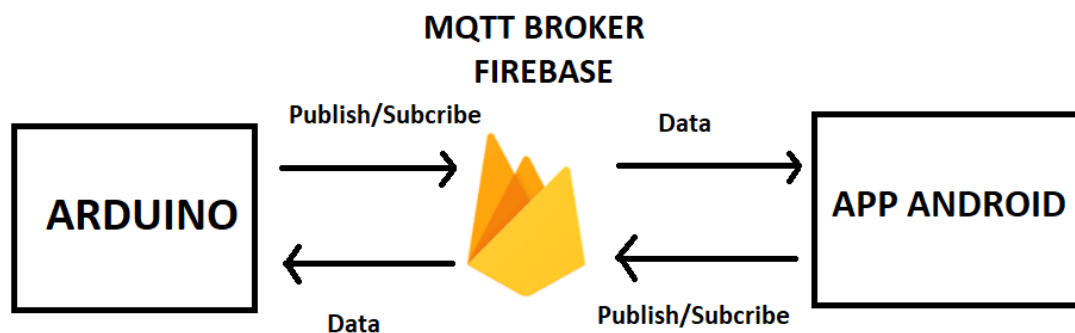
- Phân phối thông tin hiệu quả
- Tăng khả năng mở rộng
- Giảm đáng kể tiêu thụ băng thông mạng
- Giảm tốc độ cập nhật
- Phù hợp cho việc điều khiển
- Tối đa hoá băng thông đang sử dụng
- Chi phí đầu tư cực thấp
- Độ an toàn cao vì bảo mật dựa trên sự cấp phép
- Giảm thiểu thời gian phát triển

#### **b) Nhược điểm**

- MQTT có chu kì truyền chậm hơn tương đối so với CoAP
- Tài nguyên của MQTT hoạt động dựa trên Subscriber động
- MQTT là giao thức truyền thông nhưng khó để tạo ra được mạng mở rộng toàn cầu

#### **4.2.4. Ứng dụng giao thức MQTT trong đề tài**

Trong phạm vi đề tài, nhóm ứng dụng giao thức MQTT theo mô hình:



*Hình 4. 5. Mô hình ứng dụng giao thức MQTT trong đề tài*

Broker được nhóm lựa chọn là dịch vụ Firebase đã được giới thiệu.

Quá trình giao tiếp xuất phát từ App Android thực hiện điều khiển các thiết bị trong gia đình bằng mạng wifi sẽ Publish các thông tin (là các dữ liệu dạng String) tới Broker Firebase, tại đây các thông tin được lưu trữ bởi mô hình dạng cây và tiếp tục chờ để được xử lý. Khi này, Server sau khi Subscriber sẽ nhận được các dữ liệu từ Broker (là các thông tin đã được lưu trữ trước đó) và thực hiện điều khiển các thiết bị trong gia đình theo chính yêu cầu được người dùng lựa chọn trên App Android.

### **4.3. Mã hoá dữ liệu**

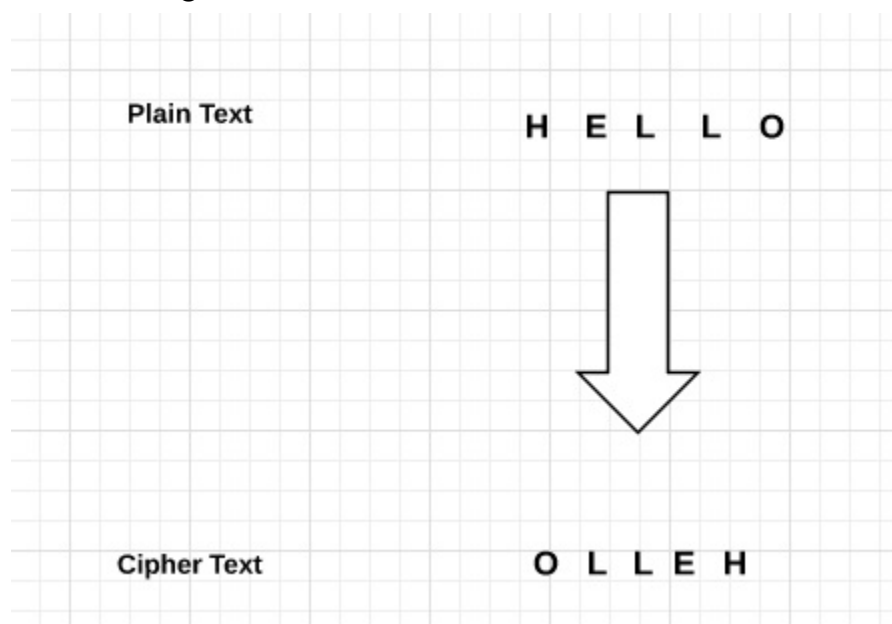
#### **4.3.1. Encryption và Decryption**

##### **a. Thuật toán Reverse Cipher**

Thuật toán Reverse Cipher có các đặc điểm:

- Sử dụng một mẫu đảo ngược của chuỗi văn bản gốc như là mật mã.

- Quá trình mã hóa và giải mã giống nhau.
- Để giải mã người dùng chỉ cần đảo ngược lại văn bản đã mã hóa để thu được văn bản gốc.

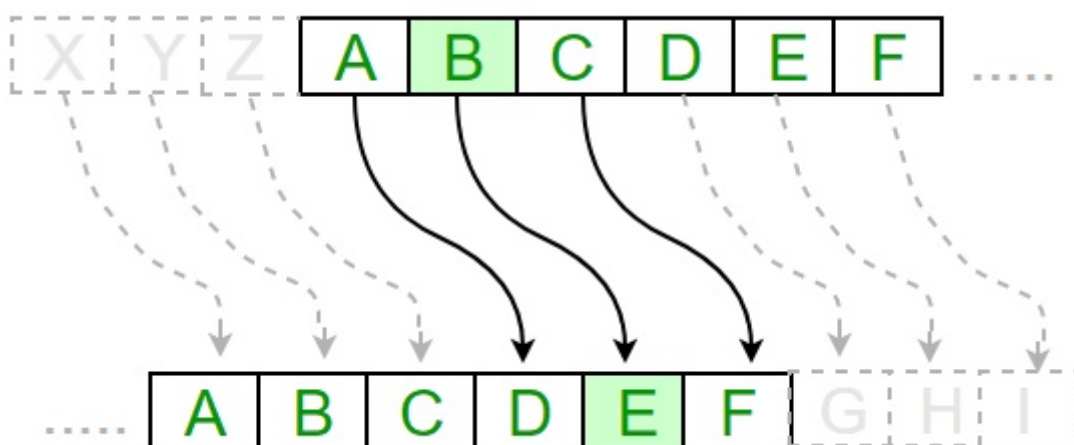


Hình 4. 6. Reverse Cipher

Thuật toán Reverse Cipher khá đơn giản nên mức độ bảo mật yếu, có thể dễ dàng bị giải mã để thu được văn bản gốc, do đó ít được áp dụng để mã hóa dữ liệu quan trọng.

#### b. Thuật toán Caesar Cipher

- Thuật toán Caesar Cipher có các đặc điểm
- Thuật toán khá đơn và dễ dàng thực hiện.
- Mỗi chữ cái sẽ được thay thế bằng chữ cái khác trong bảng chữ cái theo quy luật.



Hình 4. 7. Mã hóa Ceasar Cipher

Với mỗi key sẽ xác định số bước tịnh tiến kí tự thành kí tự tiếp theo để mã hóa đơn văn bản. Để giải mã áp dụng tương tự quá trình mã hóa với cùng key nhưng ngược chiều.

ROT13 là một trường hợp đặc biệt của thuật toán Caesar Cipher với key là 13. Do trong bảng chữ cái tiếng Anh có 26 kí tự nên với key là 13 các kí tự sẽ có tính đối xứng

A	B	C	D	E	F	G	H	I	J	K	L	M
↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕	↕
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Hình 4. 8. Mã hóa ROT13

### c. Thuật toán Transposition Cipher

Các thuật toán Transposition Cipher sử dụng các phương pháp nhằm sắp xếp lại các ký tự trong văn bản gốc để tạo thành văn bản mới được mã hóa. Ví dụ như phương pháp thay columnar transposition cipher, với mỗi key sẽ xác định một bảng điều ký tự có chiều rộng bằng key. Các ký tự trong văn bản lần lượt được điền vào bảng theo hàng ngang, văn bản đã mã hóa là chuỗi ký tự được đọc lần lượt theo từng hàng dọc. Với key = 4.

h	e	l	l
o	w	o	r
l	d		

Hình 4. 9. Bảng mã hóa

Văn bản “helloworld” được mã hóa thành “holewdlrlr”

Có một phương pháp khác là sử dụng vị trí của các ký tự trong văn bản gốc. Thuật toán sẽ khởi tạo “key” chuỗi rỗng và được đánh số từ 0 tới “key”. Sau đó tiến hành lấy dư của vị trí các ký tự tương ứng trong văn bản gốc và đưa vào chuỗi có đánh số tương ứng, cuối cùng ghép lại được chuỗi đã mã hóa.

T	n	o	t	n	p	r
r	s	s	i	C	h	
a	p	i	o	i	e	

Hình 4. 10. Bảng mã hóa

Với Key = 3, chuỗi “TranspositionCipher” được mã hoá thành “TnotnprssiChapioie”

Để mã hóa ngược chỉ cần đọc các ký tự theo cột của bảng mã hóa trên.

Việc lựa chọn key của thuật toán Transposition Cipher có vai trò quan trọng, nếu key có giá trị là 0 văn bản mã sẽ bị bỏ trống, nếu key có giá trị là 1 hoặc có giá trị lớn hơn độ dài văn bản gốc thì văn bản mã hóa sẽ giống văn bản gốc. Nhưng nếu lựa chọn key nhỏ (khác 1 và 0) thì văn bản mã hóa sẽ dễ bị đoán.

#### d. Thư viện Cryptography

Thư viện Cryptography sử dụng thuật toán XOR, thuật toán XOR tiến hành xor bit từng ký tự trong văn bản gốc với key để đưa ra văn bản đã được mã hóa.

Thuật toán XOR có các đặc điểm:

- Tốc độ tính toán nhanh.
- Quá trình mã hóa và giải mã là giống nhau với cùng key (Tính 2 chiều)

Trong khi sử dụng kỹ thuật mã hóa Caesar, việc mã hóa và giải mã các ký hiệu liên quan đến việc chuyển đổi các giá trị thành số sau đó sử dụng phép cộng hoặc trừ rồi lấy dư để tính ký tự sau khi được mã hóa.

0	1	2	3	4	5	6	7	8	9	10	11	12
A	B	C	D	E	F	G	H	I	J	K	L	M
13	14	15	16	17	18	19	20	21	22	23	24	25
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Hình 4. 11. Bảng quy đổi

Ví dụ, với key = 4, mã hóa của A là  $(0 + 4) \% 26 = 4$  (E).

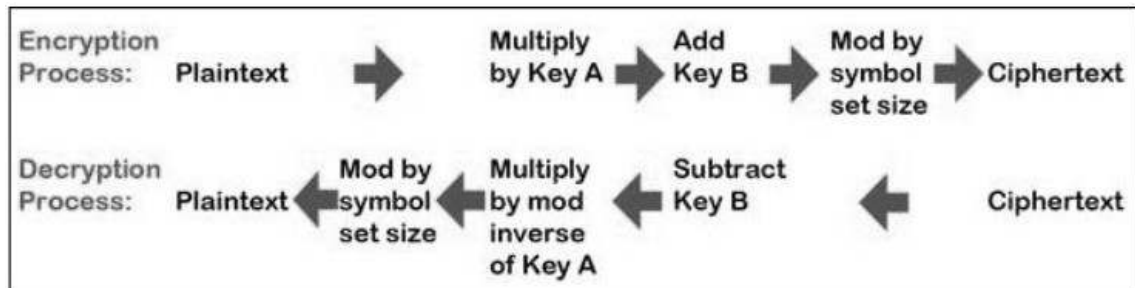
Tuy nhiên thay vì sử dụng với phép cộng hoặc trừ như kỹ thuật mã hóa Ceasar, có thể sử dụng Multiplicative Cipher. Đối với mỗi ký tự sẽ được quy đổi thành số trong bảng trên, sau đó tiến hành nhân với key và lấy số dư khi chia cho 26.

Plaintext Symbol	Number	Encryption with Key	Ciphertext Symbol
A	0	$(0 * 7) \% 26 = 0$	A
B	1	$(1 * 7) \% 26 = 7$	H
C	2	$(2 * 7) \% 26 = 14$	O
D	3	$(3 * 7) \% 26 = 21$	V
E	4	$(4 * 7) \% 26 = 2$	C
F	5	$(5 * 7) \% 26 = 9$	J
G	6	$(6 * 7) \% 26 = 16$	Q
H	7	$(7 * 7) \% 26 = 23$	X
I	8	$(8 * 7) \% 26 = 4$	E
J	9	$(9 * 7) \% 26 = 11$	L
K	10	$(10 * 7) \% 26 = 18$	S
L	11	$(11 * 7) \% 26 = 25$	Z
M	12	$(12 * 7) \% 26 = 6$	G
N	13	$(13 * 7) \% 26 = 13$	N
O	14	$(14 * 7) \% 26 = 20$	U
P	15	$(15 * 7) \% 26 = 1$	B
Q	16	$(16 * 7) \% 26 = 8$	I
R	17	$(17 * 7) \% 26 = 15$	P
S	18	$(18 * 7) \% 26 = 22$	W
T	19	$(19 * 7) \% 26 = 3$	D
U	20	$(20 * 7) \% 26 = 10$	K
V	21	$(21 * 7) \% 26 = 17$	R
W	22	$(22 * 7) \% 26 = 24$	Y



X	23	$(23 * 7) \% 26 = 5$	F
Y	24	$(24 * 7) \% 26 = 12$	M
Z	25	$(25 * 7) \% 26 = 19$	T

Hình 4. 12. Thuật toán Affine Cipher trong thư viện Cryptography sẽ kết hợp thuật toán Multiplicative Cipher và Caesar Cipher



Hình 4. 13. Thuật toán Affine Cipher

Ngoài ra một số thuật toán khác như Monoalphabetic Cipher (thay thế một ký tự bởi một ký tự khác trong danh sách), Simple Substitution Cipher (thay thế ký tự bằng ký tự khác trong văn bản gốc), ...

Thư viện Cryptography cung cấp đa dạng các thuật toán mã hóa và các interface of coding ở mức cao giúp người dung dễ dàng mã hóa bảo vệ tài liệu, văn bản quan trọng, qua đó giảm thiểu nguy cơ mất an toàn thông tin.

#### 4.3.2. Mã hoá dữ liệu SHA – 512

SHA-512 dựa trên hàm băm có các đặc điểm:

- Tính xác định: một input cho ra một output duy nhất
- Dễ tính toán: tính nhanh chóng
- Không thể giải ngược
- Tránh va chạm: Xác suất hai văn bản cho ra cùng output có điểm chung là cực thấp
- Hiệu ứng nở tuyết
- Đa dạng dữ liệu truyền vào

##### a. Giới thiệu thuật toán SHA

Năm 1993 NIST (National Institute of Standards and Technology) công bố thuật toán mã hóa có tên Secure Hash Standard (SHA-0), sau đó bị thu hồi bởi NSA sau khoảng thời gian ngắn và công bố một phiên bản thay thế là SHA-1. Đây là một hàm băm mật mã giúp tạo ra đoạn mã 160-bit từ văn bản gốc.

Từ năm 2005, SHA-1 không còn được coi là an toàn trước các mối nguy, do đó tới năm 2010 nhiều tổ chức khuyến cáo nên thay thế. NIST chính thức không sử dụng SHA-1 từ năm 2011 và không cho phép sử dụng thuật toán này trong chữ ký số từ năm 2013.

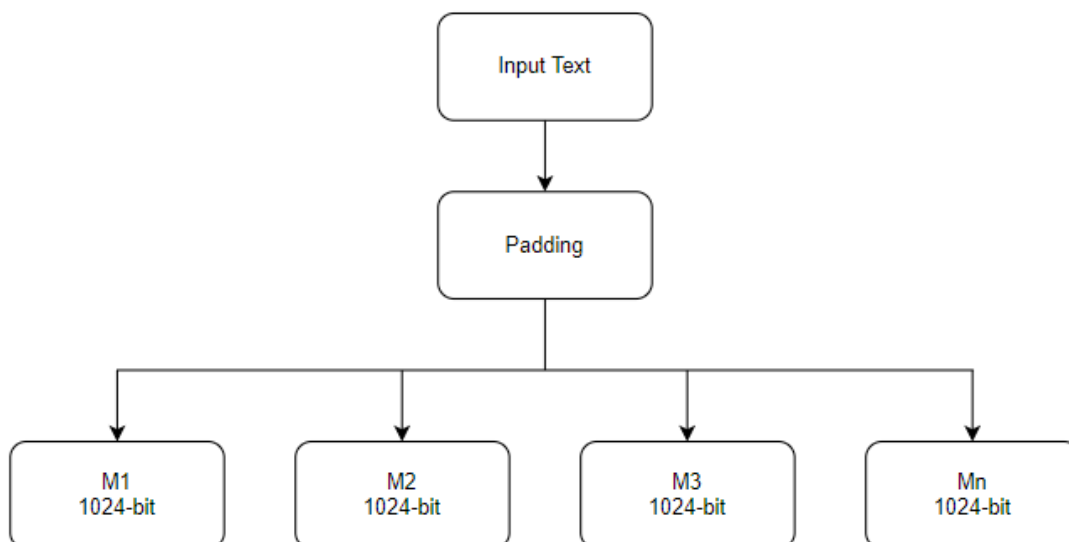
Sau đó phiên bản SHA-2 công bố với nhiều thay đổi từ phiên bản tiền nhiệm SHA-1. SHA-2 gồm 6 hàm băm với giá trị băm là 224, 256, 384 và 512 bit: SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256. Trong đó SHA-512 được nhiều tổ chức khuyến cáo dung bởi tính bảo mật cao, tuy



nhien SHA-512 có cách tính phức tạp hơn nên thời gian thực thi lâu hơn các thuật toán còn lại.

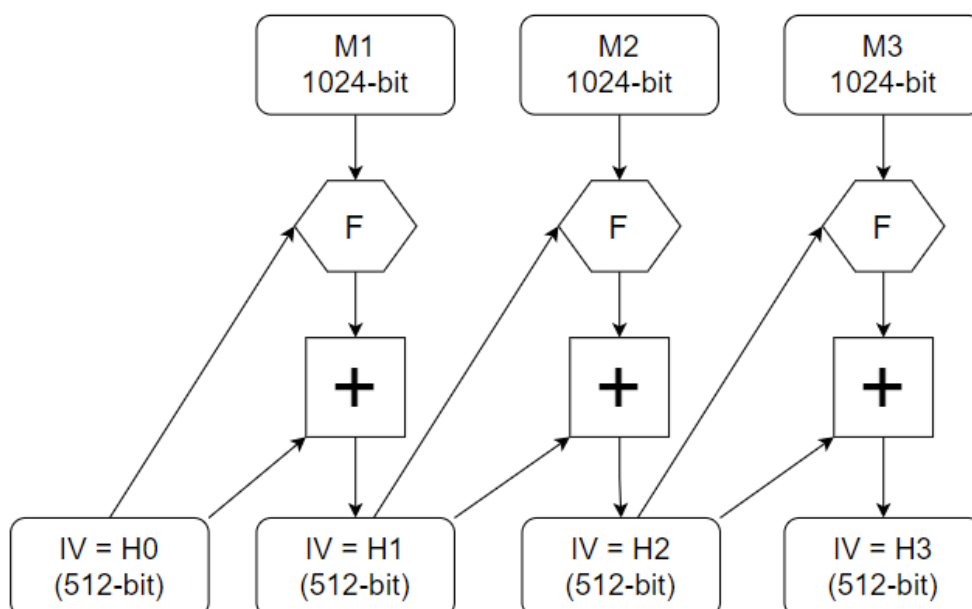
### b. Thuật toán SHA-512

Thuật toán SHA-512 nhận giá trị đầu vào là đoạn văn bản bất kỳ sau và trả về đoạn mã có độ dài cố định là 512-bit



Hình 4. 14. Sơ đồ thuật toán

Đoạn mã sau khi đi qua bước padding đoạn văn bản gốc sẽ được chuyển thành đoạn mã có độ dài là số chia hết cho 1024 bằng cách thêm các số “0” vào đoạn mã gốc để tổng số bit khi chia cho 1024 là 896, 128-bit cuối sẽ biểu diễn độ dài của văn bản gốc. Sau đó tiến hành chia đoạn mã thành n phần có độ dài 1024-bit.



Hình 4. 15. Sơ đồ khối tổng thể

Các đoạn mã 1024-bit ở trên lần được thực hiện theo sơ đồ, Khối đầu tiên sẽ được truyền vào module F kết hợp với khối có độ dài 512-bit (Khối này được tạo ra từ 8 Block 64-bit là A, B, C, D, E, F, G, H được quy định trước). Sau đó

cho ra kết quả truyền tiếp vào khối addition, ở đây khối tiếp tục được truyền vào sau khi qua bước đảo bit và XOR bit. Cuối cùng thực hiện phép lấy dư cho để thu được khối và thực hiện tiếp tục đối với các khối tiếp theo.

A = 0x6a09e667f3bcc908;

B = 0xbb67ae8584caa73b;

C = 0x3c6ef372fe94f82b;

D = 0xa54ff53a5f1d36f1;

E = 0x510e527fade682d1;

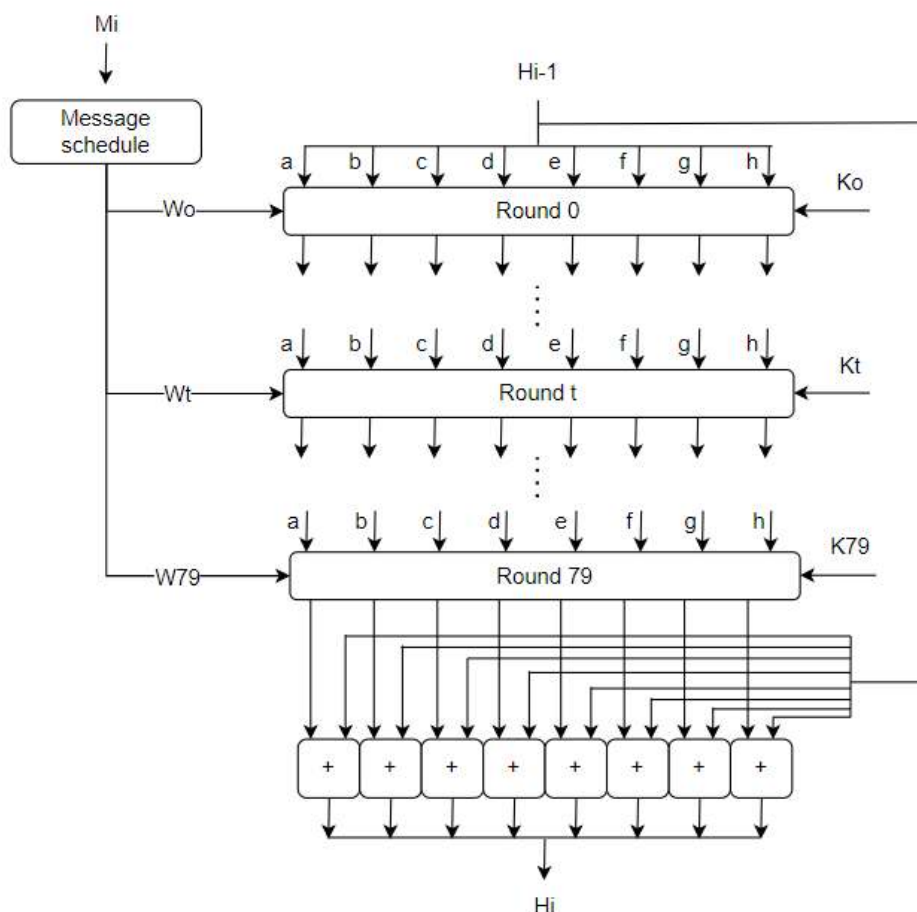
F = 0x9b05688c2b3e6c1f;

G = 0x1f83d9abfb41bd6b;

H = 0x5be0cd19137e2179;

Khối F module ở trên tiến hành chia khối M có độ dài 1024-bit thành các Word là mỗi Word có độ dài 64-bit. Với mỗi khối M độ dài 1024-bit chia thành 16 phần ứng với  $k$  từ sẽ được tính theo công thức:

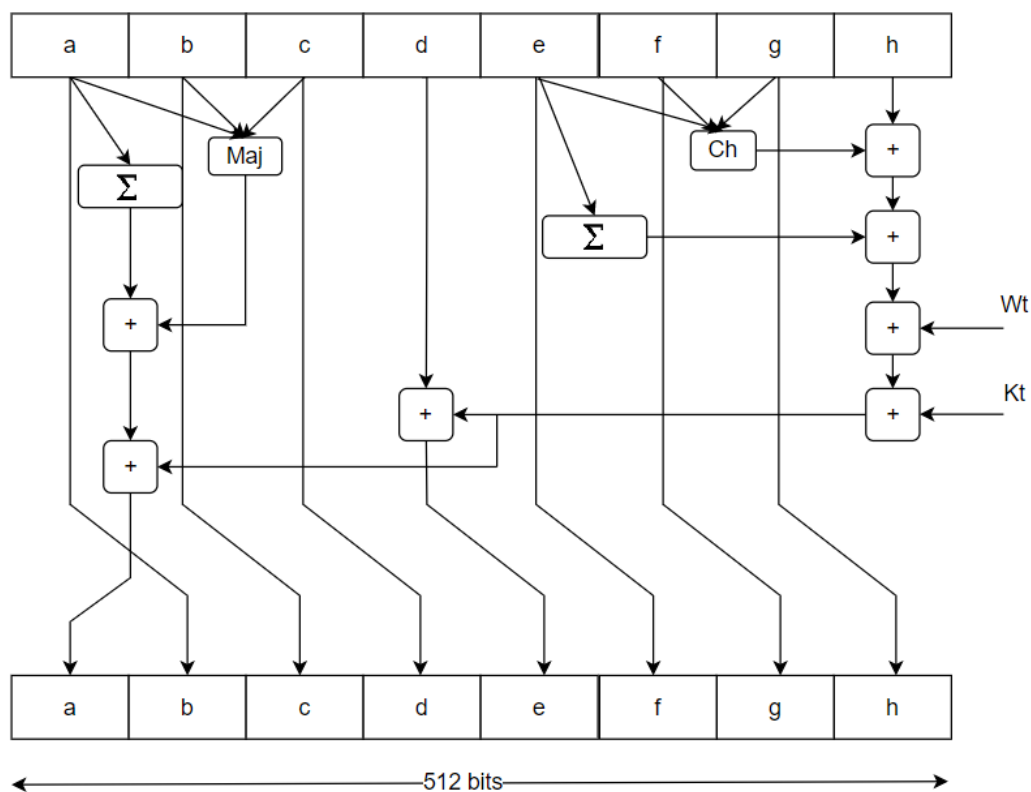
Toán tử + ở đây được hiểu là cộng số dư của biến với



*Hình 4. 16. Quá trình xử lý mỗi block 1024-bit*

Các Word sẽ lần lượt tham gia vào tính toán ở các Round tương ứng, bên cạnh đó có sự tham gia của các hằng số và các hằng số a, b, c, d, e, f, g, h được khởi tạo dựa trên block .

Các hằng số có độ dài 64-bit được định nghĩa là phân thập phân của căn bậc hai của 80 số nguyên tố đầu tiên



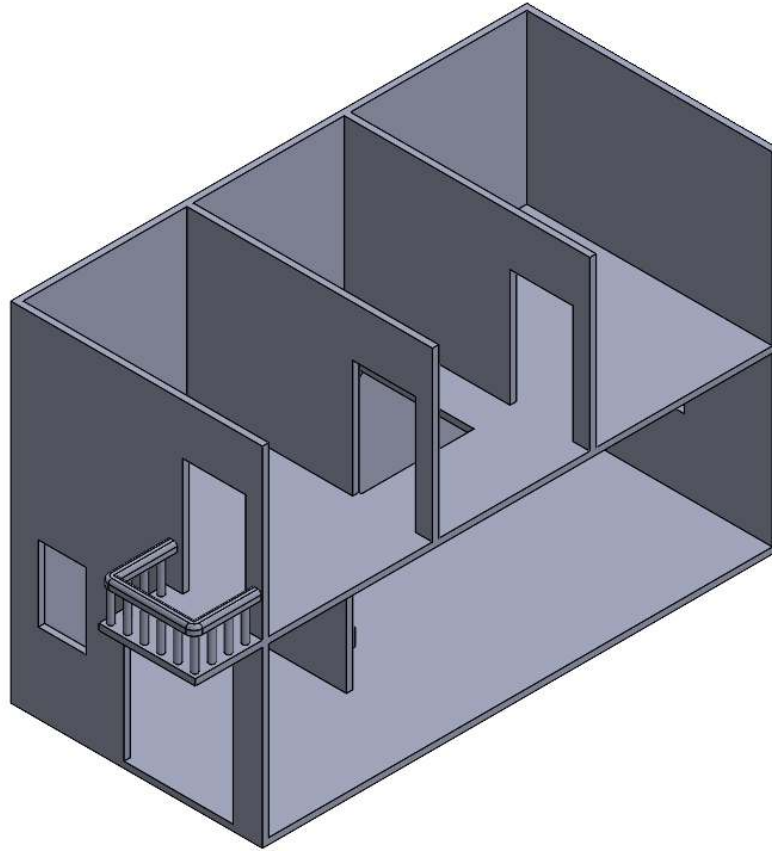
Hình 4. 17. Sơ đồ hoạt động mỗi round

Do đó:

Sau khi qua các round, các đoạn mã được tổng hợp để đưa ra mã đã được mã hóa

## CHƯƠNG 5. THIẾT KẾ MÔ HÌNH NHÀ THÔNG MINH

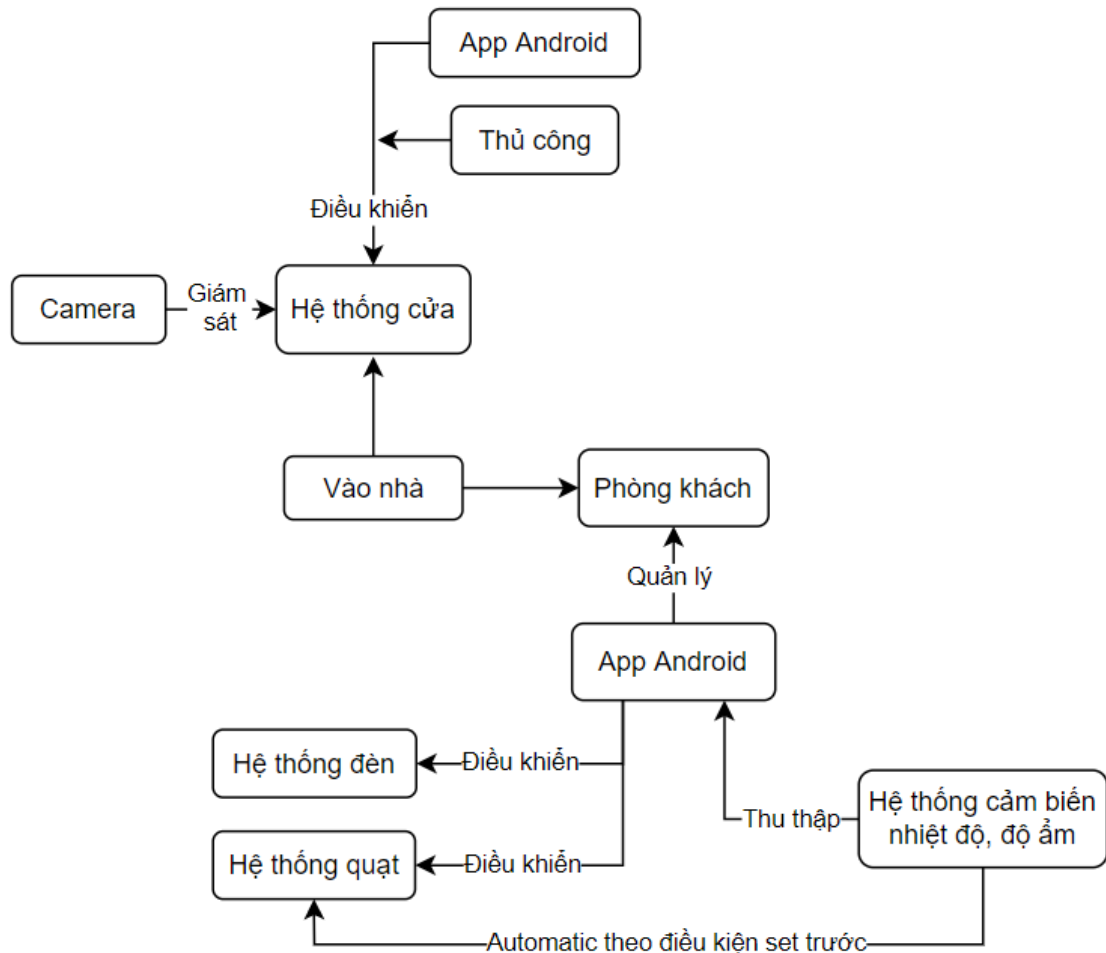
### 5.1. Sơ đồ kết cấu của ngôi nhà



*Hình 5. 1. Sơ đồ mô hình nhà thông minh*

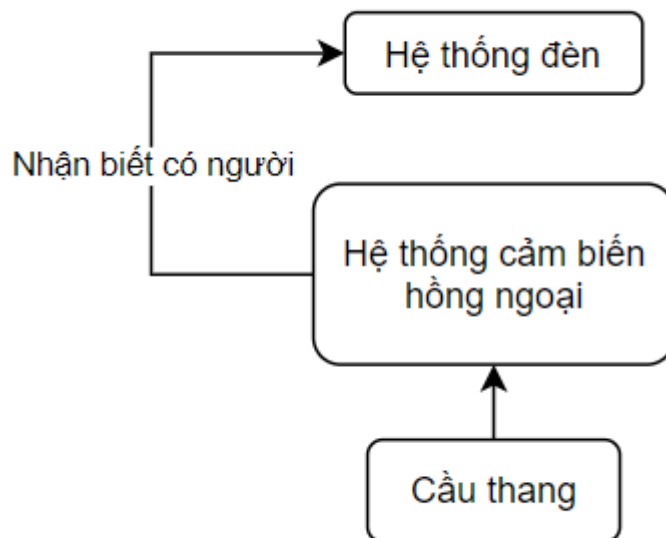
### 5.2. Nguyên lý hoạt động của mô hình nhà thông minh

### 5.2.1. Sơ đồ nguyên lý chung



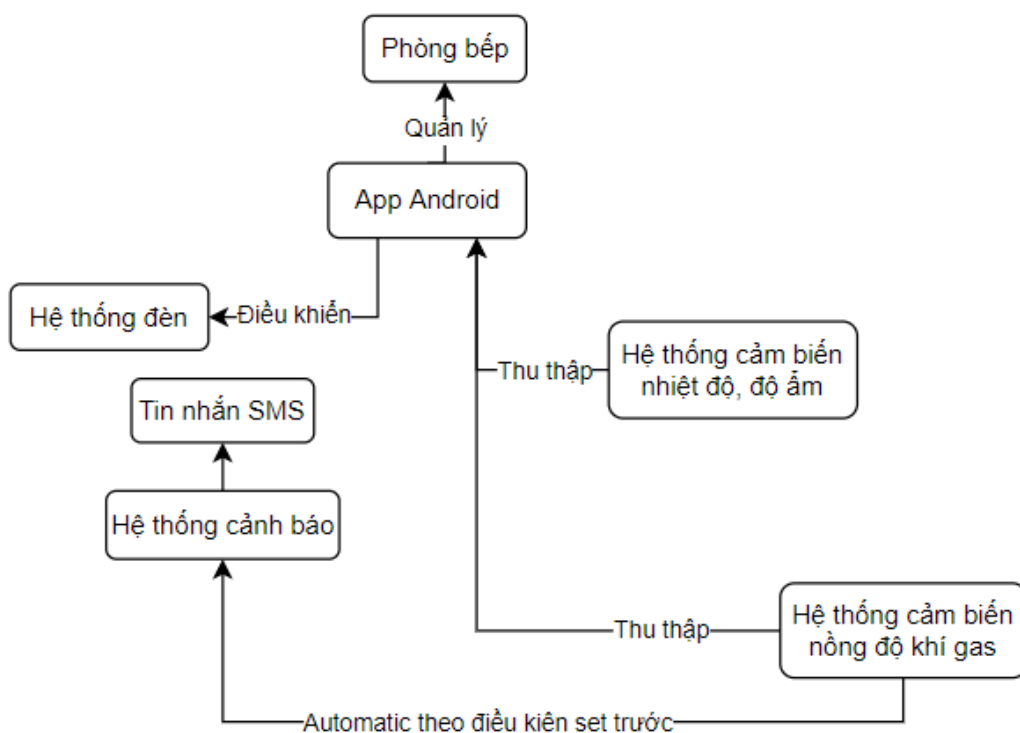
Hình 5. 2. Trường hợp người dùng phòng khách

Khu vực cửa chính ở phòng khách được đóng/mở thực hiện qua app điện thoại của người sử dụng. Khi đi vào khu vực phòng khách, người dùng có thể thực hiện bật/tắt đèn và quạt thông qua app điện thoại. Tuy nhiên, nếu nhiệt độ trong phòng khách quá cao (so với một giá trị được set up trước) thì quạt sẽ tự động bật để làm mát cho phòng.



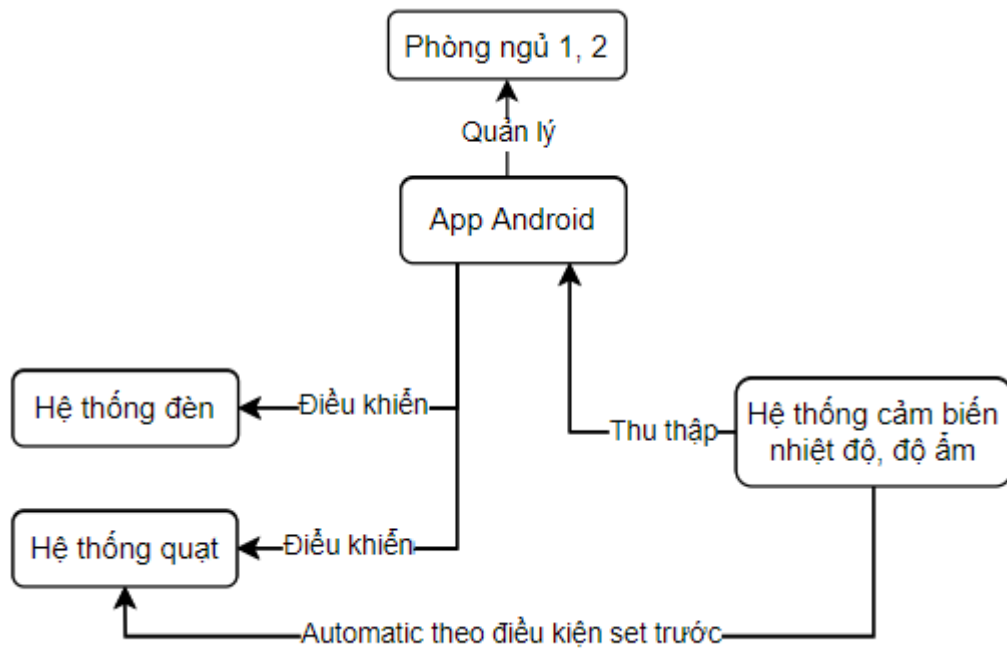
*Hình 5. 3. Trường hợp người dùng khu vực cầu thang*

Khu vực cầu thang, được lắp đặt một cảm biến hồng ngoại nhằm phát hiện có sự xuất hiện của con người, nếu có sẽ thực hiện bật đèn chiếu sáng cho người bước đi trên cầu thang.



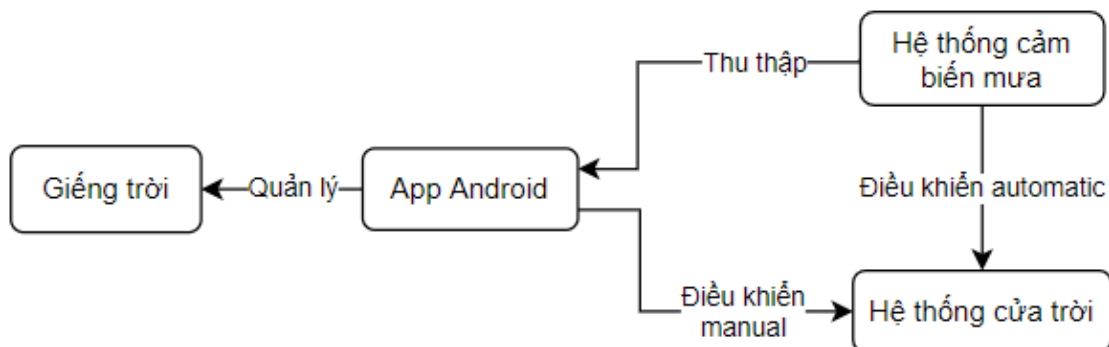
*Hình 5. 4. Trường hợp người dùng phòng bếp*

Khu vực phòng bếp, được trang bị thiết bị chiếu sáng, cảm biến nhiệt độ và nồng độ cồn để cảnh thông báo tới app android và cảnh báo đến tin nhắn người dùng có dấu hiệu của hỏa hoạn.



*Hình 5. 5. Trường hợp người dùng các phòng ngủ*

Các khu vực phòng ngủ được trang bị thiết bị chiếu sáng, cảm biến nhiệt độ và hệ thống quạt với chức năng tương tự đối với phòng khách.

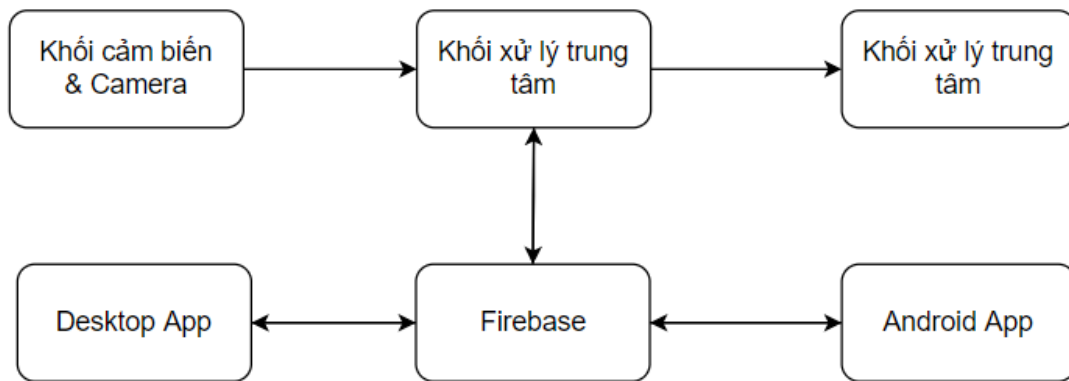


*Hình 5. 6. Trường hợp người dùng khu vực giếng trời*

Khu vực giếng trời được trang bị cảm biến mưa nhằm phát hiện tình trạng mưa ở khu vực hiện nhà ở. Nếu có mưa sẽ cảnh báo tới app người dùng, từ đó người dùng có thể thực hiện đóng/mở giếng trời kịp lúc.



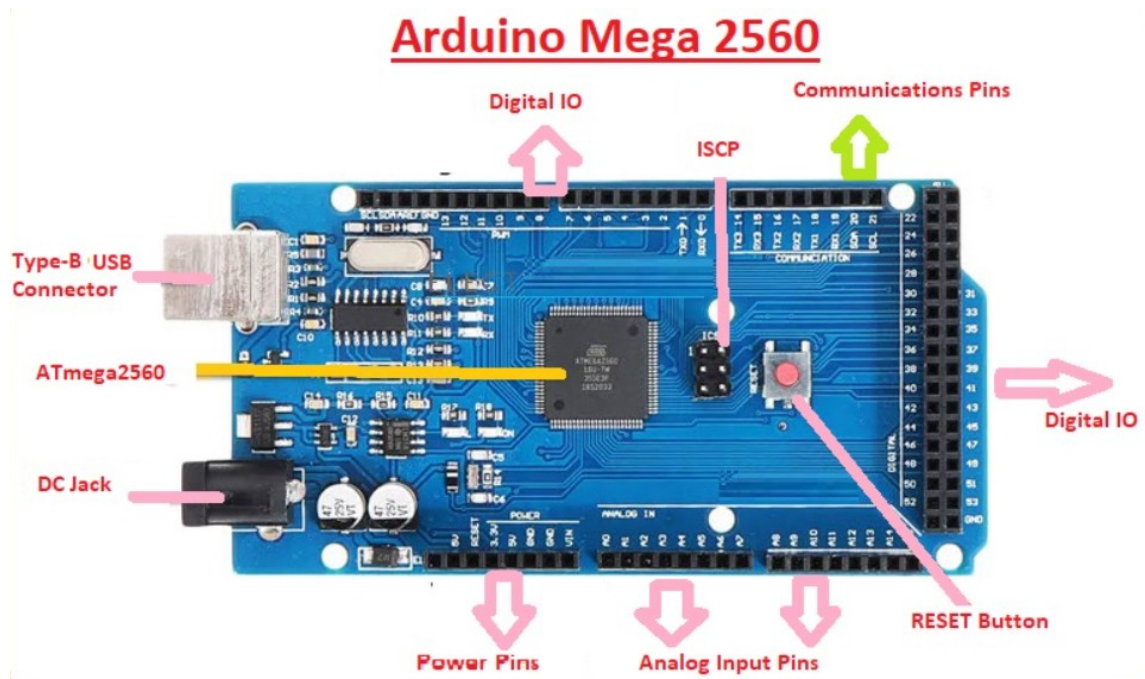
### 5.2.2. Sơ đồ khối hệ thống và chức năng



Hình 5. 7. Sơ đồ khối hoạt động của toàn bộ mô hình

### 5.3. Lựa chọn linh kiện và module trong mạch điện tử

#### 5.3.1. Module vi điều khiển Arduino Mega 2560



Hình 5. 8. Vi điều khiển Arduino Mega 2560

#### a) Thông số kỹ thuật

- Vi điều khiển: Arduino Mega 2560 sử dụng chip Atmega2560
- 54 chân digital (15 chân có thể sử dụng được như các chân PWM)
- 16 chân đầu vào analog
- 4 cổng UARTs (cổng nổi tiếng phân cứng)
- 1 thạch anh 16MHz
- Nguồn cung cấp 7-12V
- Bộ nhớ flash 128KB
- SRAM 8KB

- Giao tiếp: USB (Lập trình Atmega8), ICSP, SPI, I2C và USART
- Bộ timer 6: 2 (8bit) + 4(16bit)

#### **b) Lý do lựa chọn**

- Arduino Mega 2560 có bộ flash lớn, số chân nhiều hơn so với các dòng Arduino Uno, Arduino Nano..., cùng số lượng shield hỗ trợ không hề nhỏ nên Arduino Mega 2560 rất phù hợp được dùng cho các dự án lớn khi phải xử lý nhiều luồng thông tin, đặc biệt là các dự án IoT.
- Bộ nhớ flash của Arduino Mega được tăng lên một cách đáng kể, gấp 4 lần so với dòng Arduino Uno R3. Việc trang bị 3 timer và 6 cổng interrupter khiến Arduino Mega 2560 có thể xử lý song song nhiều luồng dữ liệu số cũng như tương tự.
- Lập trình cho vi điều khiển Arduino Mega 2560 tương đối đơn giản, phù hợp cho người học nghiên cứu các mô hình có ứng dụng Arduino vào hệ thống IoT.

### **5.3.2. Module cảm biến nhiệt độ, độ ẩm DHT11**

#### **a) Cảm biến nhiệt độ, độ ẩm DHT11**

DHT11 là một cảm biến kỹ thuật số để cảm nhận nhiệt độ, độ ẩm. Cảm biến này có thể dễ dàng giao tiếp với bất kỳ bộ vi điều khiển nào như Arduino, Raspberry Pi... để đo độ ẩm và nhiệt độ ngay tức khắc. Cảm biến sử dụng một điện trở nhiệt và một cảm biến độ ẩm điện dung để xác định độ ẩm tương đối xung quanh.

#### **b) Cấu tạo của DHT11**

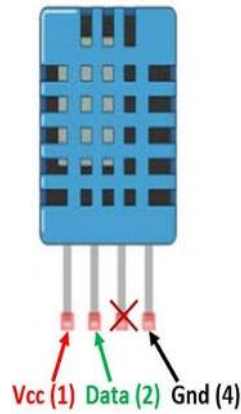
Cảm biến DHT11 bao gồm một phần tử cảm biến độ ẩm điện dung và một điện trở nhiệt cảm nhận nhiệt độ. Tự điện cảm biến độ ẩm có hai điện cực với chất nền giữ ẩm làm chất điện môi giữa chúng. Thay đổi giá trị điện dung xảy ra với sự thay đổi của các mức độ ẩm. IC đo, xử lý các giá trị điện trở đã thay đổi này và chuyển chúng thành dạng kỹ thuật số.

Để đo nhiệt độ, cảm biến sử dụng một nhiệt điện trở có hệ số nhiệt độ âm, làm giảm giá trị điện trở của nó khi nhiệt độ tăng. Để có được giá trị điện trở lớn hơn ngay cả đối với sự thay đổi nhỏ nhất của nhiệt độ, cảm biến thường được làm bằng gốm bán dẫn hoặc polymer.

#### **c) Sơ đồ chân của DHT11**

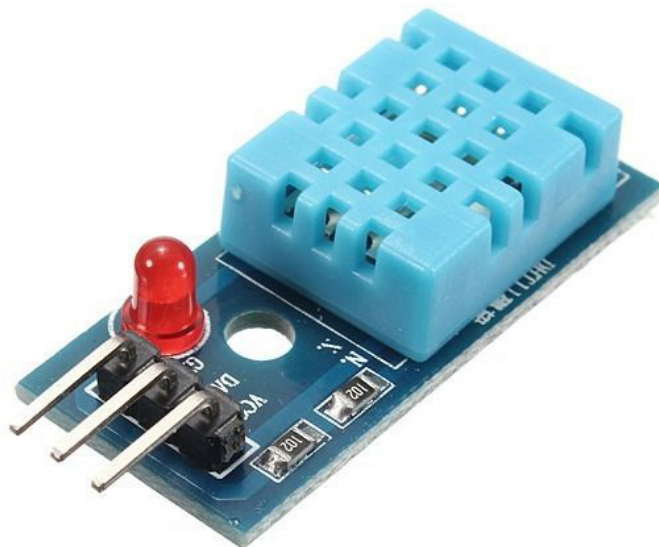
*Bảng 1. Sơ đồ nối chân của DHT11*

STT	Tên chân	Mô tả
1	Vcc	Nguồn nuôi từ 3.5V đến 5.5V
2	Data	Đầu ra là nhiệt độ, độ ẩm thông qua dữ liệu nối tiếp
3	NC	Không có kết nối, do đó không sử dụng
4	Ground	Chân nối đất



*Hình 5. 9. Cảm biến DHT11*

Cảm biến DHT11 gồm 4 chân tuy nhiên chỉ sử dụng 3 chân gồm: Vcc, Data và GND do đó cảm biến thường được tích hợp thành một module cảm biến có 3 chân: Vcc, Data và GND tiện lợi, chắc chắn.



*Hình 5. 10. Module cảm biến DHT11 được nhóm sử dụng*

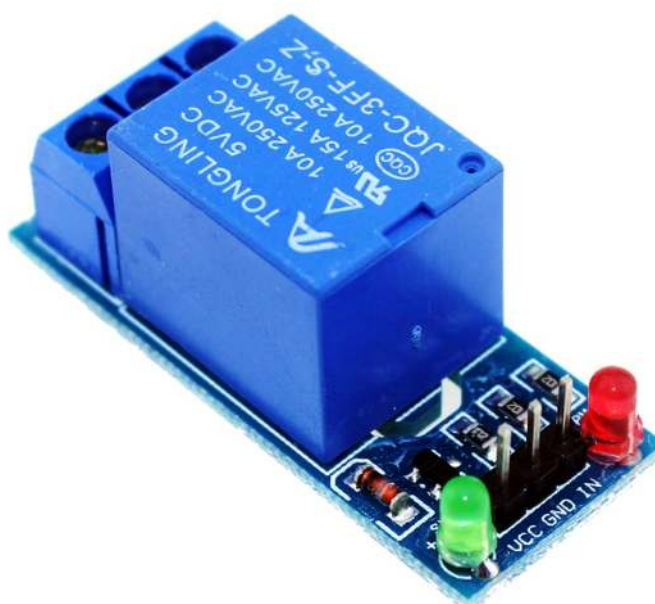
#### **d) Thông số kỹ thuật của DHT11**

- Điện áp hoạt động: 3.5V đến 5.5V
- Dòng điện tiêu thụ: 2.5 mA
- Phạm vi cảm biến độ ẩm: 20% - 90% RH, sai số 5% RH
- Phạm vi cảm biến nhiệt độ: 0 độ - 50 độ, sai số 2 độ
- Tần số lấy mẫu tối đa: 1Hz (1 giây 1 lần)
- Kích thước: 23\*12\*5 mm

### 5.3.3. Relay 1 kênh 5V kích mức thấp

#### a) Lý do lựa chọn

- Relay 1 kênh 5V được nhóm lựa chọn sử dụng để điều khiển đóng/mở thiết bị ngoại vi quạt tại các phòng.
- Module sử dụng loại Relay tốt, đảm bảo hoạt động ổn định, dài lâu.
- Trên module có opto để cách ly dòng ngược về, hiệu suất ổn định.
- Có thể set các mức cao thấp bằng cách thiết lập jumper trên module.
- Có led báo nguồn của relay (màu xanh), led báo trạng thái hoạt động của Relay (màu đỏ).
- Việc kết nối giữa module relay với mạch điều khiển vô cùng đơn giản.



Hình 5. 11. Relay 1 kênh 5V kích mức thấp

#### b) Thông số kỹ thuật

Bảng 2. Thông số kỹ thuật của Relay sử dụng

Thông số	Giá trị
Điện áp tải tối đa	AC 250V-10A / DC 30V-10A
Điện áp điều khiển	5 VDC
Dòng kích Relay	5 mA
Trạng thái kích	Mức thấp (0V)
Đường kính lỗ ốc	3.1 mm
Kích thước	50 x 26 x 18.5(mm)

#### c) Cấu tạo của relay

Module relay 1 kênh sử dụng chân kích mức thấp (0V), khi có tín hiệu vào chân IN thì relay sẽ chuyển qua trạng thái “thường hở” của Relay. Module relay có tiếp điểm đóng ngắt gồm 3 tiếp điểm: NC (thường đóng), NO (thường mở) và COM (chân chung) được cách ly hoàn toàn với board mạch chính. Ở trạng thái

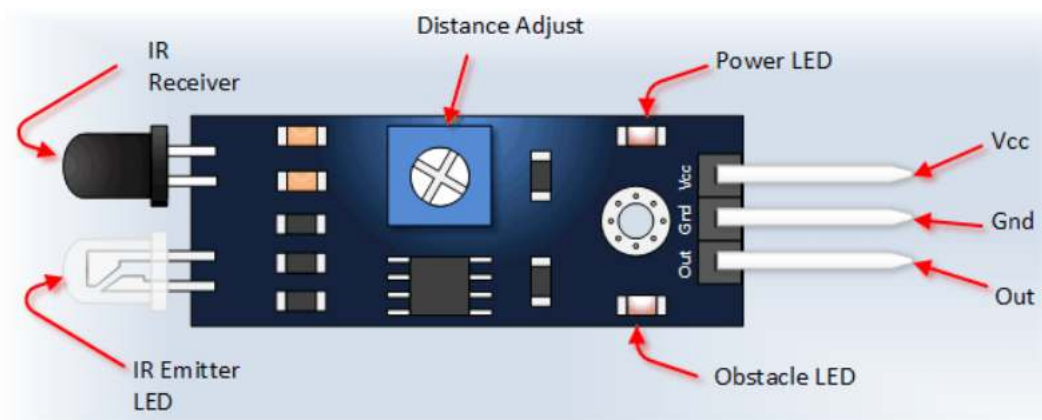
bình thường, chưa kích NC sẽ nối với COM, khi có trạng thái kích COM sẽ chuyển sang nối với NO và mất kết nối với NC.

**d) Sơ đồ chân của Relay**

STT	Tên chân	Mô tả

**5.3.4. Module cảm biến hồng ngoại IR**

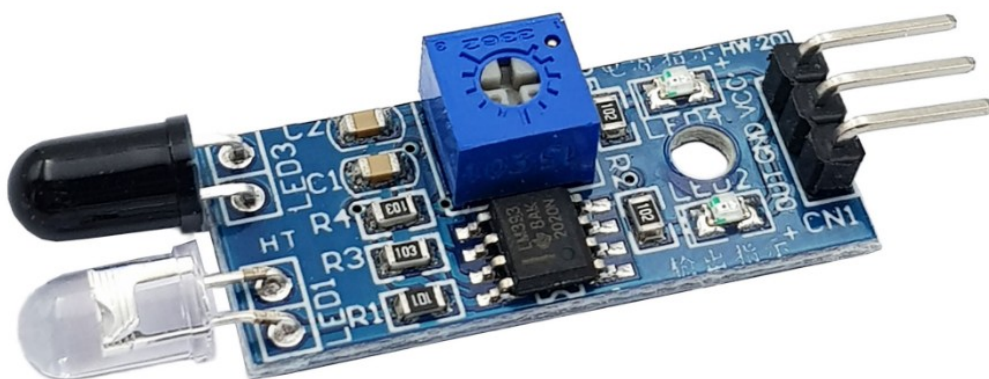
**a) Sơ đồ chân của cảm biến hồng ngoại IR**



*Hình 5. 12. Module cảm biến hồng ngoại IR*

**Cổng giao tiếp:**

- VCC: điện áp chuyển đổi từ 3.3V đến 5V (có thể kết nối trực tiếp với vi điều khiển 5V và 3.3V)
- GND: chân Ground
- OUT: chân đầu ra digital (0 và 1)



*Hình 5. 13. Module cảm biến hồng ngoại được nhóm sử dụng*

**b) Thông số kỹ thuật**

- Bộ so sánh sử dụng LM393, hoạt động ổn định, liên tục
- Điện áp làm việc: 3.3V đến 5V DC
- Khi cấp nguồn cho IR, đèn báo nguồn (màu đỏ) sáng
- Lỗ vít 3mm



- Kích thước 3.2 x 1.4 (cm)
- Các module đã được so sánh điện áp ngưỡng thông qua chiết áp

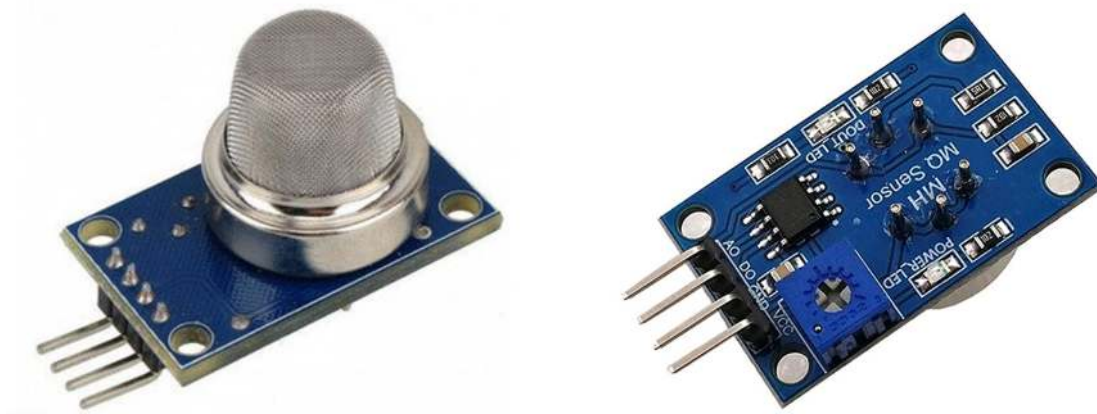
### c) Lý do lựa chọn

- Cảm biến vật cản hồng ngoại có khả năng thích nghi với môi trường, có một cặp truyền, nhận tia hồng ngoại. Tia hồng ngoại phát ra một tần số nhất định, khi phát hiện hướng truyền có vật cản (mặt phản xạ), phản xạ vào đèn thu hồng ngoại, sau khi so sánh, đèn màu xanh sẽ sáng lên, đồng thời cho tín hiệu số đầu ra (một tín hiệu bậc thấp).
- Khoảng cách làm việc hiệu quả từ 2cm đến 5cm, phù hợp với mô hình nhỏ. Điện áp làm việc 3.3V đến 5V. Độ nhạy sáng của cảm biến vật cản hồng ngoại được điều chỉnh bằng chiết áp nên có thể dễ dàng điều chỉnh.

### 5.3.5. Module cảm biến nồng độ khí gas MQ – 6

#### a) Lý do lựa chọn

- Module cảm biến nồng độ khí gas MQ-6 là một module có ứng dụng cao, rộng rãi, có khả năng phát hiện các khí dễ cháy nổ ở nồng độ thấp, sử dụng tốt trong các ứng dụng cảnh báo và chữa cháy, giúp phát hiện các sự cố rò rỉ khí gas và có thể phát hiện được một số loại khí độc (propane, butane...) ở nồng độ thấp.
- Module hỗ trợ cả đầu ra số và đầu ra tương tự, có chiết áp điều chỉnh độ nhạy cho đầu ra số. Khi mà nồng độ khí phát hiện càng cao thì điện áp trên chân đầu ra A0 càng lớn (tối đa là 4.5V).
- Tốc độ phản hồi nhanh, độ nhạy cao, mạch thiết kế đơn giản có thể sử dụng lâu dài mà vẫn giữ được tính ổn định.

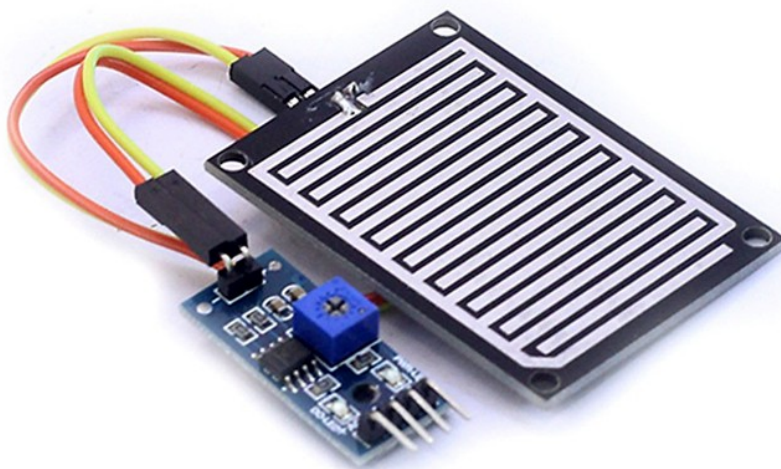


*Hình 5. 14. Module cảm biến nồng độ khí gas MQ-6*

#### b) Thông số kỹ thuật

- Nguồn hoạt động: 5V
- Module sử dụng LM393 và MQ-6
- Ổn định khi sử dụng trong thời gian dài
- Kích thước: 32.4mm x 20.4 mm, khối lượng 6,85 gram

### 5.3.6. Module cảm biến mưa



*Hình 5. 15. Module cảm biến mưa*

#### **a) Thông số kỹ thuật**

- Kích thước tấm cảm biến mưa: 54 x 40mm
- Kích thước board PCB: 30 x 16mm
- Điện áp hoạt động: 5V
- Đáp ứng cả đầu ra kỹ thuật số (0 và 1) và đầu ra tương tự (điện áp A0)
- Lỗ cố định, bu lông dễ dàng lắp đặt
- Có đèn báo hiệu nguồn và đầu ra
- Đầu ra TTL, tín hiệu đầu ra TTL có giá trị thấp. Có thể điều khiển được Relay, buzzer,...
- Độ nhạy có thể được điều chỉnh thông qua chiết áp
- LED sáng lên khi không có mưa, LED tắt khi có mưa

#### **b) Sơ đồ chân kết nối**

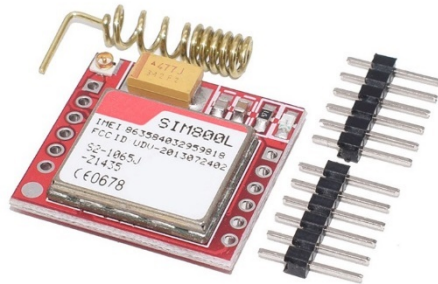
- VCC: chân nguồn
- GND: chân nối đất
- D0: đầu ra tín hiệu digital
- A0: đầu ra tín hiệu analog

#### **c) Cấu tạo, nguyên lý hoạt động của module cảm biến mưa**

- Bộ phận nhận biết mưa được gắn bên ngoài trời
- Bộ phận điều chỉnh độ nhạy được che chắn lại
- Mạch cảm biến mưa hoạt động bằng cách so sánh hiệu điện thế của mạch cảm biến nằm bên ngoài trời với một giá trị định mức trước (giá trị này thay đổi được thông qua biến trở màu xanh được gắn với module) từ đó phát ra tín hiệu đóng/ngắt relay qua chân D0. Khi cảm biến khô ráo (tức trời không mưa), chân D0 của module sẽ được giữ ở mức cao (5V – 12V). Khi có xuất hiện nước trên bề

mặt cảm biến (tức trời đang mưa), đèn LED màu đỏ sẽ sáng lên, chân D0 được kéo xuống mức thấp (0V).

### 5.3.7. Module Sim – 800L



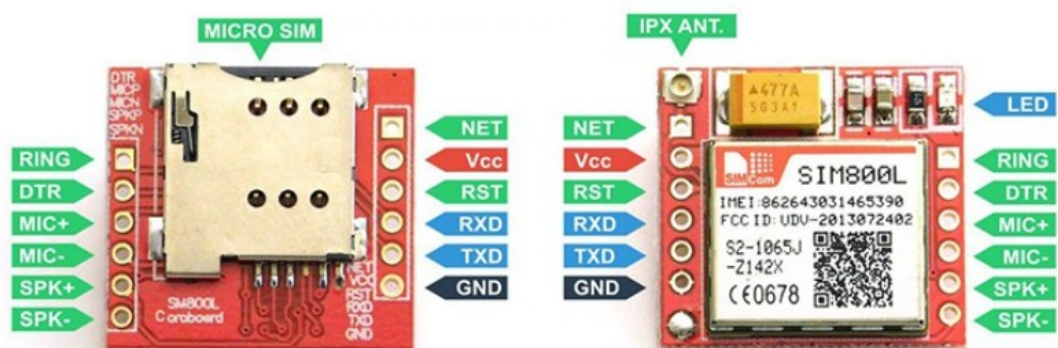
Hình 5. 16. Module nghe gọi nhắn tin Sim - 800L

Module GSM GPRS sim800L có khả năng nhắn tin SMS, nghe, gọi, GPRS... như một điện thoại nhưng có kích thước nhỏ nhất trong các loại module sim (25mm x 22mm). Điều khiển module sử dụng tập lệnh AT dễ dàng, chân kết nối dùng rào đực thông dụng (male header) chuẩn 100 mil.

#### a) Thông số kỹ thuật

- Nguồn cấp: 3.7V – 4.2V DC, có thể sử dụng nguồn dòng thấp từ 500 mAh trở lên (như cổng USB, nguồn từ board Arduino)
- Khe cắm sim: MICROSIM
- Dòng khi ở chế độ chờ: 10mA
- Dòng khi hoạt động: 100mA đến 1A
- Hỗ trợ 4 băng tần phổ biến
- Kích thước 25mm x 22mm

#### b) Sơ đồ chân



Hình 5. 17. Sơ đồ chân của module Sim - 800L

- VCC: Nguồn vào 3.7 – 4.2 VDC
- TXD: Chân truyền đạt Uart TX



- RXD: Chân truyền đạt Uart RX
- DTR: Chân UART DTR thường không sử dụng
- SPKP, SPKN: ngõ ra âm thanh, nối với loa để phát ra âm thanh
- MICP, MICN: ngõ vào âm thanh, phải gắn thêm micro để thu âm thanh
- Reset: Chân khởi động lại sim800L thường không sử dụng
- RING: Báo có cuộc gọi đến
- GND: Chân Mass, cấp 0V

### c) Lý do lựa chọn

- Module nhỏ gọn, có cải tiến hơn so với các dòng sim800A, bắt sóng tốt
- Có thể kết nối với vi điều khiển thông qua giao thức UART, điều khiển qua tập lệnh AT dễ dàng thao tác
- Có khả năng gọi điện, nhắn tin, nhận tin nhắn... ổn định, hiệu quả

### d) Tập lệnh lập trình AT

#### Các lệnh chung:

- Lệnh: AT<CR><LF>

Mô tả: Kiểm tra đáp ứng của Module Sim 900A, nếu trả về OK thì Module hoạt động

- Lệnh: ATE[x]<CR><LF>

Mô tả: Chế độ echo là chế độ phản hồi dữ liệu truyền đến của module Sim 900A, x = 1 bật chế độ echo, x = 0 tắt chế độ echo (bạn nên tắt chế độ này khi giao tiếp với vi điều khiển)

- Lệnh: AT+IPR=[baud rate]<CR><LF>

Mô tả: cài đặt tốc độ giao tiếp dữ liệu với Module Sim800C, chỉ cài được các tốc độ sau baud rate : 0 (auto), 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200

- Lệnh: AT+W<CR><LF>

Mô tả : lưu lại các lệnh đã cài đặt

#### Các lệnh điều khiển cuộc gọi:

- Lệnh: AT+CLIP=1<CR><LF>

Mô tả: Hiển thị thông tin cuộc gọi đến

- Lệnh: ATD[Số\_điện\_thoại];<CR><LF>

Mô tả: Lệnh thực hiện cuộc gọi

- Lệnh: ATH<CR><LF>

Mô tả: Lệnh thực hiện kết thúc cuộc gọi, hoặc cúp máy khi có cuộc gọi đến

- Lệnh: ATA<CR><LF>

Mô tả : Lệnh thực hiện chấp nhận khi có cuộc gọi đến

#### Các lệnh điều khiển tin nhắn:

- Lệnh: AT+CMGF=1<CR><LF>

Mô tả: Lệnh đưa SMS về chế độ Text , phải có lệnh này mới gửi nhận tin nhắn dạng Text

- Lệnh: AT+CMGS="Số\_điện\_thoại"<CR><LF>

Đợi đến khi có ký tự '>' được gửi về thì đánh nội dung tin nhắn

Gửi mã Ctrl+Z hay 0x1A để kết thúc nội dung và gửi tin nhắn

Mô tả: Lệnh gửi tin nhắn

- Lệnh: AT+CMGR=x<CR><LF>

x là địa chỉ tin nhắn cần đọc

Mô tả: Đọc một nhắn vừa gửi đến, lệnh được trả về nội dung tin nhắn, thông tin người gửi, thời gian gửi

- Lệnh: AT+CMGDA="DEL ALL"<CR><LF>

Mô tả: Xóa toàn bộ tin nhắn trong các hộp thư

- Lệnh: AT+CNMI=2,2<CR><LF>

Mô tả: Hiện thị nội dung tin nhắn ngay khi có tin nhắn đến

### 5.3.8. Thiết bị ngoại vi quạt



*Hình 5. 18. Thiết bị ngoại vi quạt 12V*

Thiết bị quạt với chức năng làm mát được mô phỏng bởi linh kiện quạt chip DC 12V. Linh kiện được thực hiện bật/tắt bởi module relay được giới thiệu ở trên.

Vì thiết bị sử dụng điện áp đầu vào là 12VDC nên cần có một nguồn nuôi độc lập giá trị 12V dành riêng cho các linh kiện quạt trong đồ án.

#### **Thông số kỹ thuật:**

- Điện áp 12V
- Cường độ dòng 0.3A
- Công suất 3.6W
- Tốc độ 3000 vòng/phút
- Chất liệu: Nhựa cao cấp

- Phụ kiện tương thích: Tấm bảo vệ quạt tản nhiệt
- Nhiệt độ hoạt động: 10 độ đến 70 độ C

### 5.3.9. Thiết bị ngoại vi Servo SG – 90



*Hình 5. 19. Động cơ Servo SG90*

Đồ án nhà thông minh sẽ có các hệ thống cửa ra vào, khi thực hiện hoạt động sẽ được quay quanh các bản lề do nhóm thiết kế và momen được truyền từ động cơ Servo thực hiện quay cửa.

Động cơ Servo giúp thực hiện truyền momen đạt các góc chính xác trong khoảng từ 0 đến 180 độ. Với kích thước nhỏ gọn, dễ sử dụng, servo sg90 được nhóm lựa chọn làm động cơ xoay cho các hệ thống cửa trong nhà.

#### **Thông số kỹ thuật**

- Khối lượng: 9g
- Kích thước: 22.2 x 11 x 8.32 mm
- Momen xoắn: 1.8kg/cm
- Tốc độ hoạt động: 60 độ trong 0.1 giây
- Điện áp hoạt động: 4.8V (có thể sử dụng nguồn 5V)
- Nhiệt độ hoạt động: 0 độ đến 55 độ C
- Dây đỏ là dây cấp nguồn 5V, dây nâu là dây GND, dây cam nối với xung tín hiệu từ 1ms đến 2ms.

### 5.3.10. Module mạch giảm áp LM2596

Mạch giảm áp DC – DC Buck LM2596 3A có kích thước nhỏ gọn, có khả năng giảm áp từ 30VDC xuống 1.5VDC mà vẫn đạt hiệu suất cao (92%), thích hợp ứng dụng cho các nguồn chia áp, hạ áp...



*Hình 5. 20. Module mạch giảm áp LM2596*

**a) Thông số kỹ thuật**

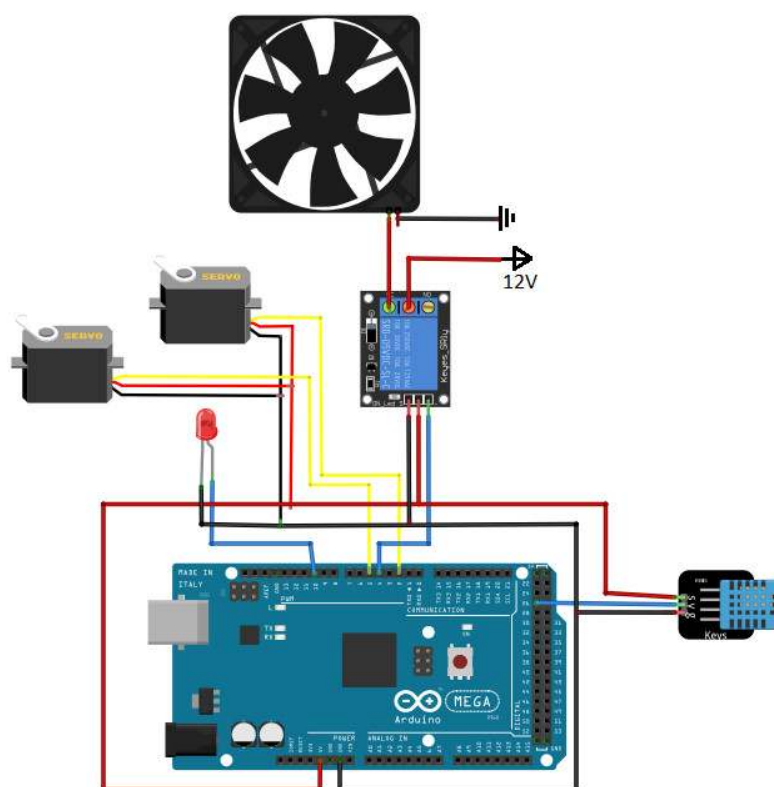
- Điện áp đầu vào: Từ 3V đến 30V
- Điện áp đầu ra: có thể điều chỉnh được trong khoảng 1.5V đến 30V
- Dòng đáp ứng tối đa là 3A, và chỉ có khả năng hạ áp
- Hiệu suất: 92%
- Kích thước: 45 x 20 x 14 mm

**b) Lý do lựa chọn**

- Điện áp đầu ra ổn định một mức duy nhất khi vận biến trở với điện áp đầu vào rộng.
- Dòng điện xả tối đa lên đến 3A

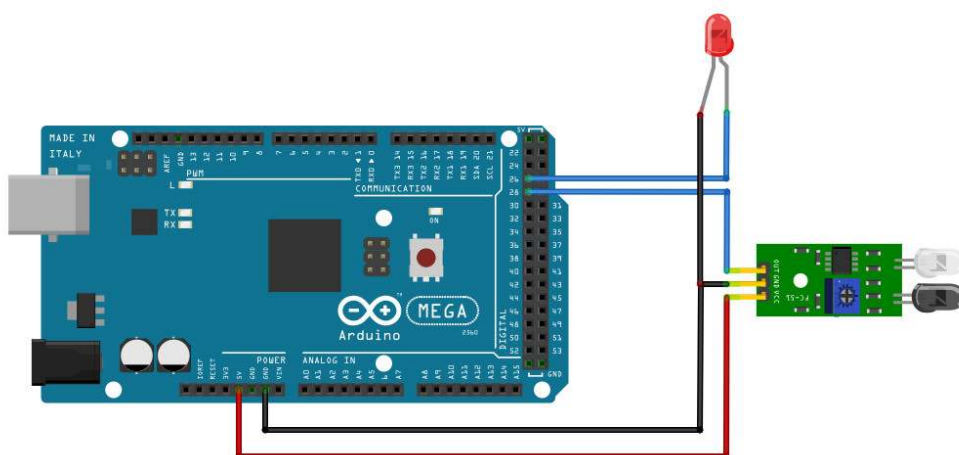
**5.4. Mạch điện trong các khu vực phòng**

### 5.4.1. Mạch điện phòng khách



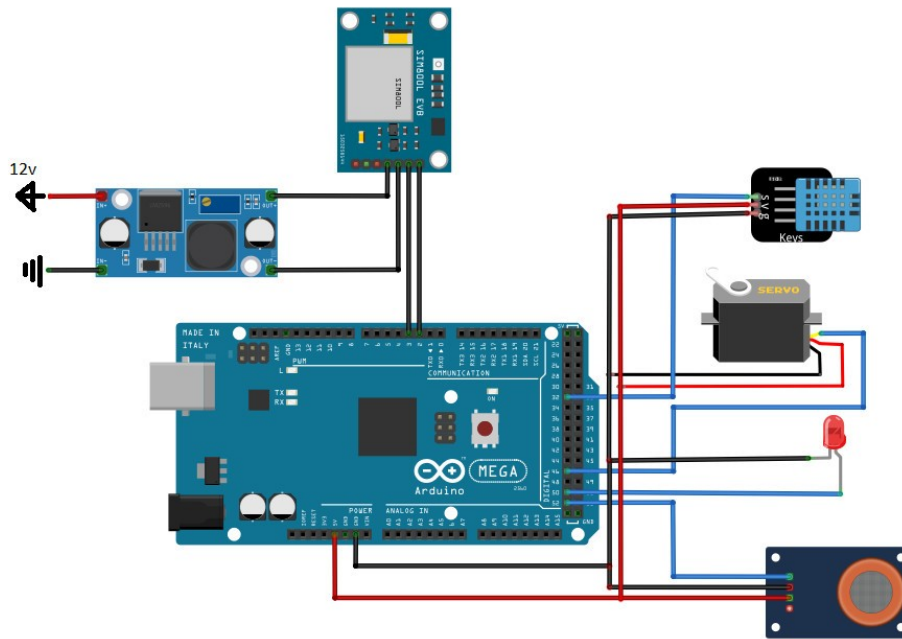
Hình 5. 21. Sơ đồ nối dây phòng khách

### 5.4.2. Mạch điện cầu thang



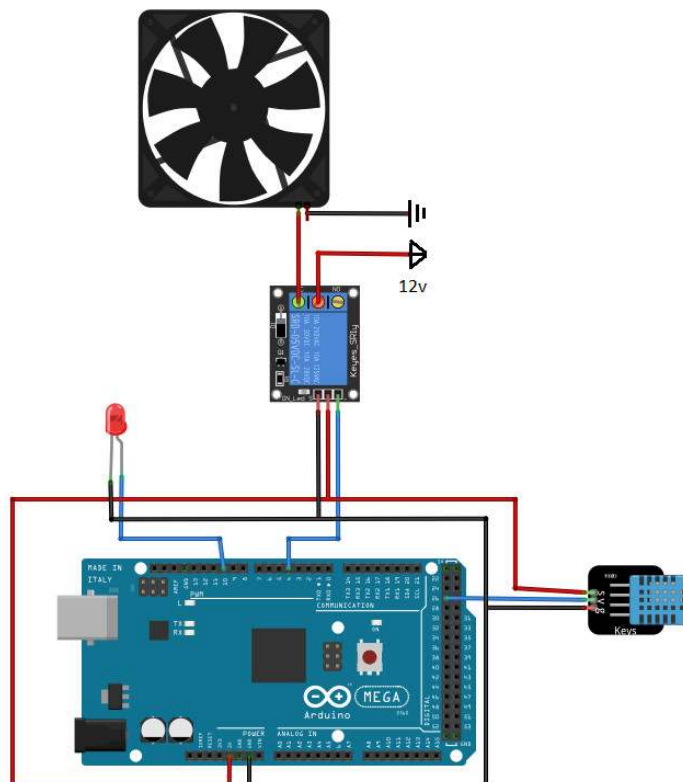
Hình 5. 22. Sơ đồ nối dây cầu thang

### 5.4.3. Mạch điện phòng bếp



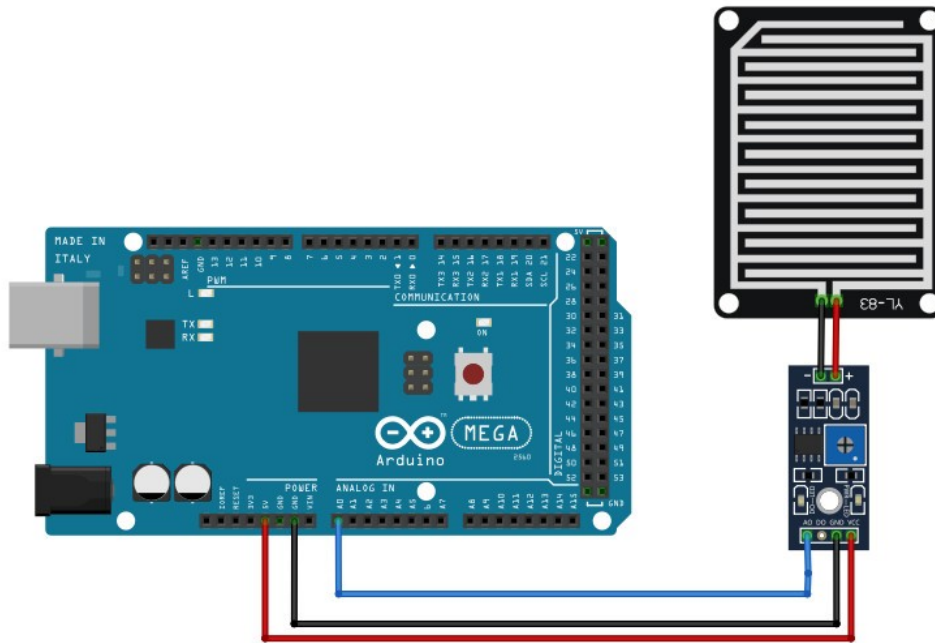
Hình 5. 23. Sơ đồ nối dây phòng bếp

### 5.4.4. Mạch điện phòng ngủ 1, 2



Hình 5. 24. Sơ đồ nối dây các phòng ngủ

#### 5.4.5. Mạch điện khu vực giếng trời



Hình 5. 25. Sơ đồ nối dây khu vực giếng trời

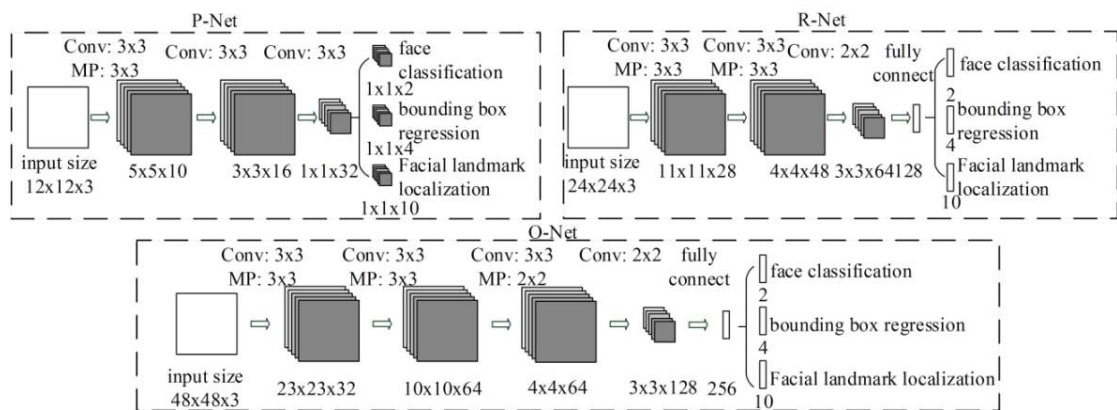
## CHƯƠNG 6. HỆ THỐNG NHẬN DIỆN KHUÔN MẶT

### 6.1. Tổng quan hệ thống

Hệ thống nhận diện khuôn mặt được ứng dụng trong việc nhận diện người ra vào nhà. Hệ thống áp dụng mô hình MTCNN để nhận diện và xác định vị trí khuôn mặt trong khung hình từ đó đưa vào mô hình FaceNet phân tích và vector hóa khuôn mặt, cuối cùng sử dụng thuật toán Support Vector Machine để phân lớp gương mặt khung hình có thuộc bộ dữ liệu không.

### 6.2. Mô hình thuật toán MTCNN

Thuật toán MTCNN (Multi-task Cascaded Convolution Neural Network) được nhóm tác giả Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, Senior Member, IEEE, and Yu Qiao, Senior Member, IEEE giới thiệu ngày 11/3/2016 giải quyết bài toán phát hiện và xác định vị trí khuôn mặt trong bức ảnh.



Hình 6. 1. Mô hình thuật toán MTCNN

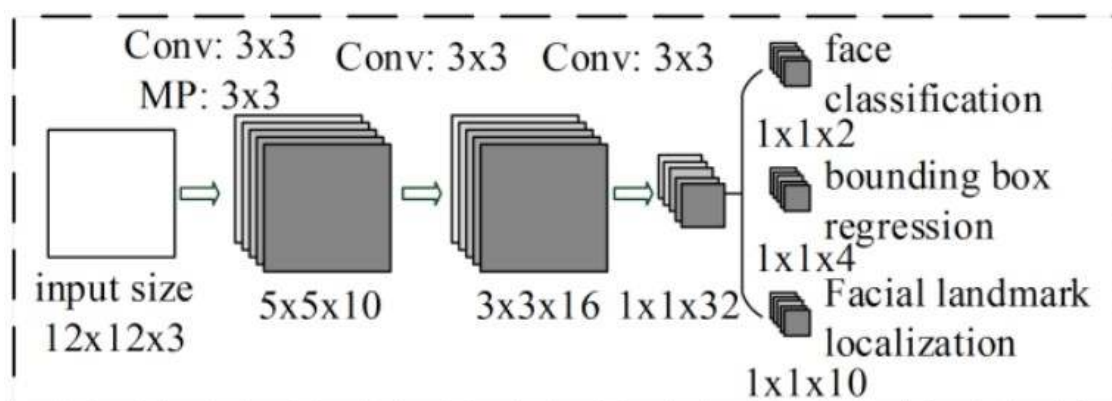
Thuật toán MTCNN bao gồm 3 mạng CNN (Convolutional Neural Networks) xếp chồng và hoạt động đồng thời khi phát hiện và xác định khuôn mặt. Mỗi mạng CNN trong MTCNN sẽ đóng vai trò khác nhau trong phát hiện khuôn mặt. Đầu vào của MTCNN là các ảnh chứa khuôn mặt cần nhận diện, thuật toán sẽ tạo ra nhiều bản sao ảnh đầu vào với nhiều tỉ lệ kích thước khác nhau để đa dạng đầu vào giúp cải thiện khả năng nhận diện.

Đầu ra của MTCNN là vector đặc trưng biểu diễn 4 tọa độ bounding box của khuôn mặt và tọa độ landmark trên mặt bao gồm 2 mắt, 1 mũi, 2 bên môi.

MTCNN hoạt động với 3 mạng riêng biệt ứng với 3 bước là P-Net, R-Net, O-Net.

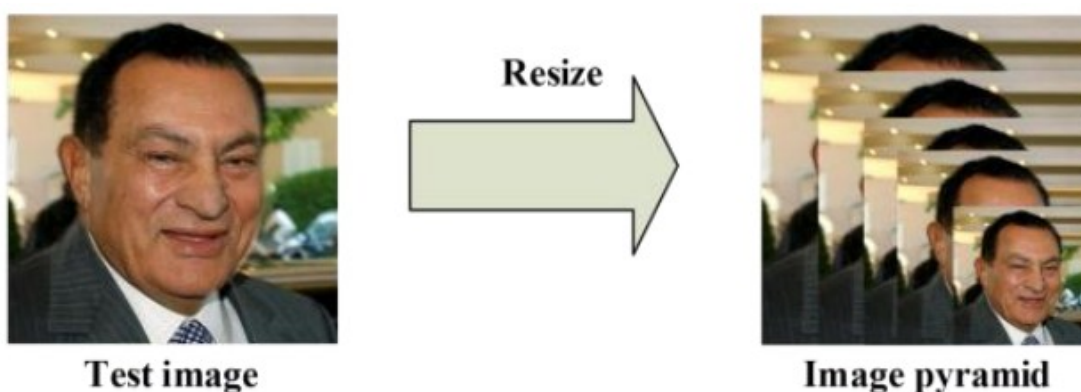


### 6.2.1. Stage 1: P-Net (Proposal Network)



Hình 6. 2. Mạng P-Net

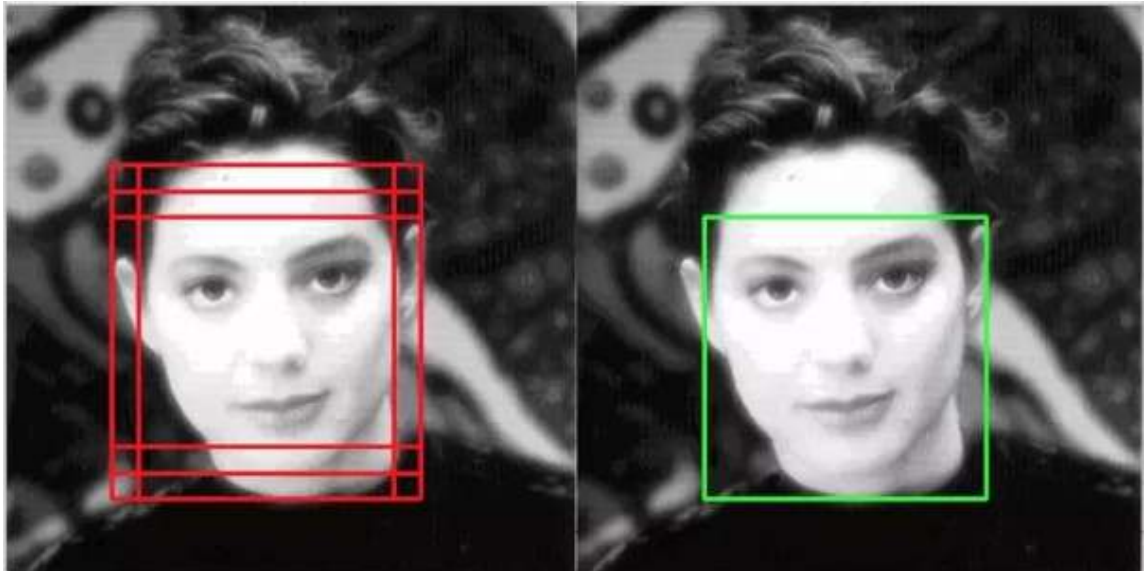
Trong một bức ảnh có thể sẽ có nhiều hơn một khuôn mặt, với kích thước đa dạng, do đó MTCNN sử dụng phương pháp thay đổi kích thước ảnh đầu vào với nhiều tỉ lệ khác nhau để tạo một loạt các bản sao chép ảnh gốc gọi là Image pyramid



Hình 6. 3. Resize ảnh

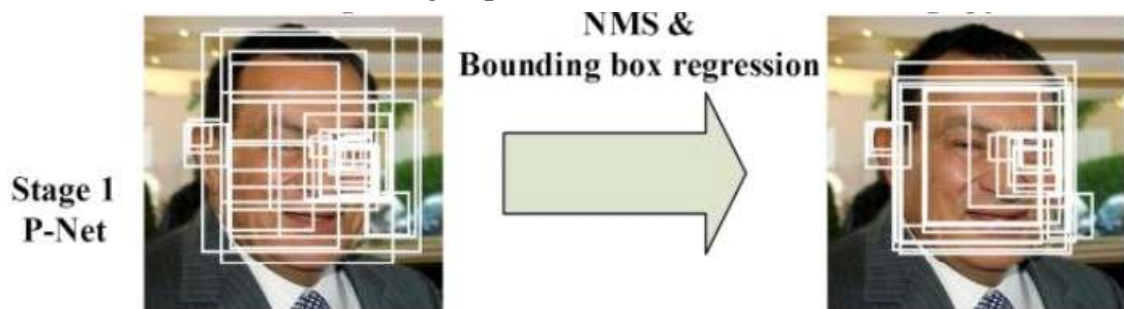
Với mỗi bản sao chép của ảnh gốc, MTCNN sử dụng kernel  $12 \times 12$  pixel và  $\text{stride} = 2$  để đi qua toàn bộ bức ảnh, dò tìm khuôn mặt. Do các bản sao chép có kích thước khác nhau, nên mạng có thể dễ dàng nhận biết được khuôn mặt với kích thước khác nhau mặc dù chỉ sử dụng 1 kernel cố định. Sau đó, đưa những kernels được cắt ra từ trên và truyền qua mạng P-Net. Kết quả sẽ cho một loạt các bounding boxes với mỗi kernel, mỗi bounding box sẽ bao gồm 4 tọa độ để định vị kernel chứa nó.

Do mỗi kernel đều có khả năng dự đoán bounding box, nên khi áp dụng ảnh với nhiều tỉ lệ khác nhau có thể dẫn tới dự đoán trùng đối tượng, để loại trừ bớt các bounding boxes trên các bức ảnh, MTCNN sử dụng 2 phương pháp là lập mức Threshold confident - nhằm xóa đi các bounding box có mức confident thấp và NMS (Non-Maximum Suppression) để xóa các box có tỷ lệ trùng nhau IoU (Intersection Over Union) vượt qua mức threshold đặt trước.



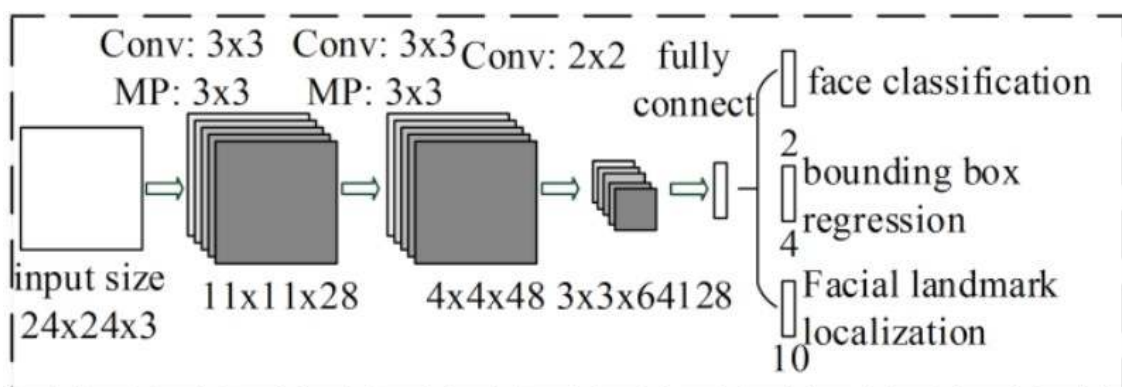
Hình 6. 4. Thuật toán Non-Maximum Suppression

Sau khi xóa các bounding box không hợp lý, tiến hành chuyển các tọa độ box về với tọa độ gốc của ảnh thật. Cuối cùng resize các box về dạng hình vuông, lấy tọa độ các box để đưa vào mạng tiếp theo.



Hình 6. 5. Minh họa Stage 1 P-Net

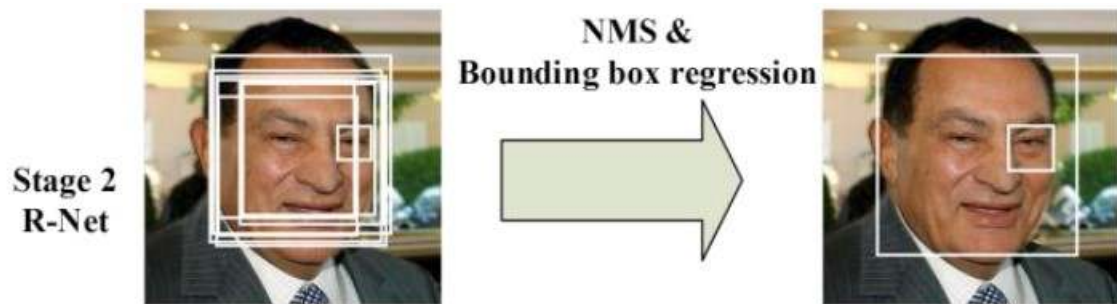
### 6.2.2. Stage 2: R-Net (Refine Network)



Hình 6. 6. Mạng R-Net

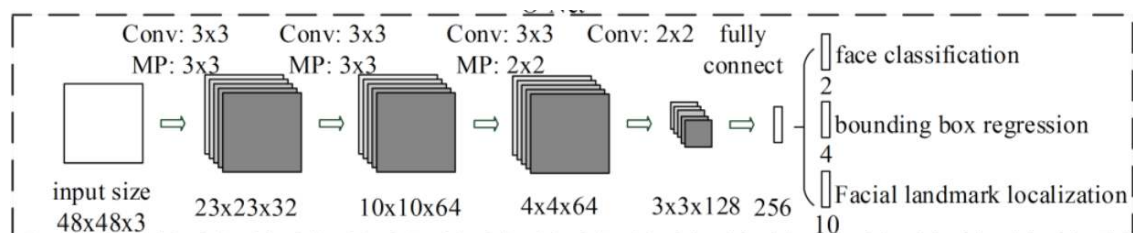
R-Net có kiến trúc tương tự như P-Net, tuy nhiên mạng sử dụng thêm padding để kích thước sau khi qua của sổ chập không thay đổi. Các bounding box ở trên sẽ được resize về kích thước 24x24, được coi như 1 kernel và được đưa vào R-Net. Mạng lọc R-Net sẽ loại bỏ một số lượng lớn các cửa sổ không chứa khuôn mặt,

sau đó hiệu chuẩn lại bounding box. Kết quả sau khi qua R-Net sẽ được resize để đưa vào O-Net.



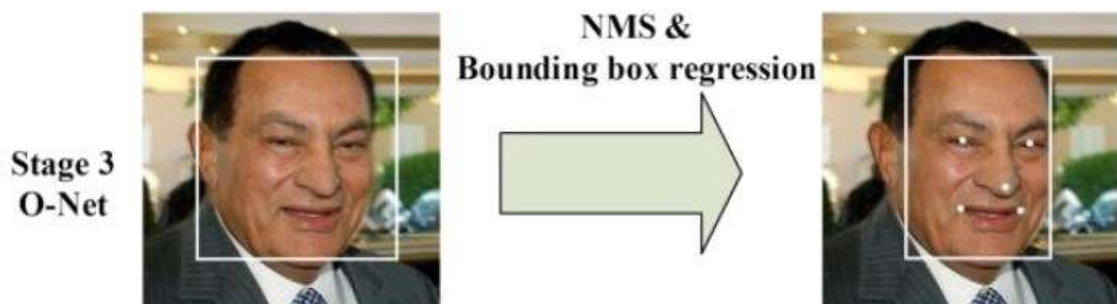
Hình 6. 7. Minh họa Stage 1 R-Net

### 6.2.3. Stage 3: O-Net (Output Network)



Hình 6. 8. Mạng O-Net

Các box đầu ra của R-Net được resize về kích thước 48x48 trước khi đưa vào mạng O-Net. Sau khi đi qua mạng O-Net sẽ dự đoán không chỉ tọa độ bounding box bao quanh khuôn mặt mà còn trả về tọa độ điểm landmark trên mặt bao gồm 2 mắt, mũi, và 2 bên môi.



Hình 6. 9. Minh họa Stage 2 O-Net

## 6.3. Mô hình thuật toán FaceNet

Facenet được nhóm kỹ sư của Google là Florian Schroff, Dmitry Kalenichenko, James Philbin giới thiệu lần đầu vào ngày 17/1/2016 trong việc nhận diện và phân lớp khuôn mặt.

### 6.3.1. Deep Convolutional Networks

Facenet được xây dựng dựa trên kiến trúc mạng GoogLeNet-Inceptionv1. Inceptionv1 đã dành chiến thắng trong cuộc thi ImageNet vào năm 2015, mạng đã giải quyết được bài toán khi đó là sử dụng kernel\_size với kích thước bao nhiêu là hợp lý.

Mạng gồm các khối Inception được xây dựng bởi 4 nhánh song song. Các bộ lọc kích thước lần lượt là **1x1**, **3x3**, **5x5** được áp dụng trong Inception Module giúp trích lọc được đa dạng đặc trưng trên những vùng nhận thức có kích thước khác nhau.

Ở đầu các nhánh 1, 2, 4 từ trên xuống, phép tích chập **1x1** được sử dụng trên từng điểm ảnh như một kết nối fully connected nhằm mục đích giảm độ sâu kênh và số lượng tham số của mô hình.

Nhánh thứ 3 từ trên xuống chúng ta giảm chiều dữ liệu bằng một layer max-pooling kích thước 3x3 và sau đó áp dụng bộ lọc kích thước 1x1 để thay đổi số kênh.

Các nhánh áp dụng padding và stride sao cho đầu ra có cùng kích cỡ chiều dài và chiều rộng. Cuối cùng ta concatenate toàn bộ kết quả đầu ra của các khối theo kênh để thu được output có kích thước bằng với input.

type	output size	depth	#1x1	#3x3 reduce	#3x3	#5x5 reduce	#5x5	pool proj (p)	params	FLOPS
conv1 (7x7x3, 2)	112x112x64	1							9K	119M
max pool + norm	56x56x64	0						m 3x3, 2		
inception (2)	56x56x192	2		64	192				115K	360M
norm + max pool	28x28x192	0						m 3x3, 2		
inception (3a)	28x28x256	2	64	96	128	16	32	m, 32p	164K	128M
inception (3b)	28x28x320	2	64	96	128	32	64	$L_2$ , 64p	228K	179M
inception (3c)	14x14x640	2	0	128	256,2	32	64,2	m 3x3,2	398K	108M
inception (4a)	14x14x640	2	256	96	192	32	64	$L_2$ , 128p	545K	107M
inception (4b)	14x14x640	2	224	112	224	32	64	$L_2$ , 128p	595K	117M
inception (4c)	14x14x640	2	192	128	256	32	64	$L_2$ , 128p	654K	128M
inception (4d)	14x14x640	2	160	144	288	32	64	$L_2$ , 128p	722K	142M
inception (4e)	7x7x1024	2	0	160	256,2	64	128,2	m 3x3,2	717K	56M
inception (5a)	7x7x1024	2	384	192	384	48	128	$L_2$ , 128p	1.6M	78M
inception (5b)	7x7x1024	2	384	192	384	48	128	m, 128p	1.6M	78M
avg pool	1x1x1024	0								
fully conn	1x1x128	1							131K	0.1M
L2 normalization	1x1x128	0								
total									7.5M	1.6B

Hình 6. 10. Kiến trúc mạng GoogLeNet – Inception v1

Các lớp Inception là tổng hợp của tất cả các lớp (1x1 conv layer, 3x3 conv layer, 5x5 conv layer) với các bộ lọc đầu ra được kết nối thành một vector đầu ra duy nhất tạo thành đầu vào cho bước tiếp theo.

### 6.3.2. Triple loss

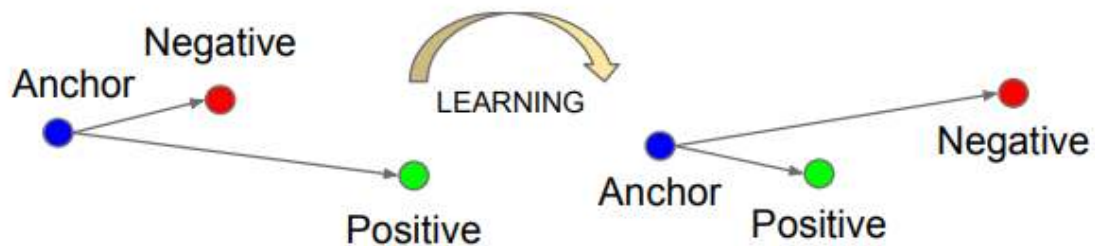
FaceNet chuyển ảnh x thành vector trong không gian Euclidean d chiều có hàm . Hàm nhúng ảnh vào không gian x chiều qua đó tính khoảng cách tới các vector ảnh khác trong không gian để thỏa mãn: một ảnh (anchor) của một người sẽ có khoảng cách gần tới những bức ảnh (positive) khác của người đó và sẽ có khoảng cách xa hơn tới những bức ảnh (negative) của người khác. Được biểu diễn bởi công thức dưới đây:

Trong đó  $\alpha$  là tham số để gia tăng khoảng cách giữa anchor với positive và negative.  $\beta$  là tất cả bộ triplets có thể tạo.

Công thức Loss khi đó:



Tuy nhiên, quá trình hội tụ trên có thể diễn ra chậm, nếu xảy ra trường hợp vector negative tới anchor gần hơn positive tới anchor, trong quá trình huấn luyện cần tinh chỉnh lại để cải thiện độ chính xác.



Hình 6. 11. Triplet Loss

Để ngăn trường hợp mô hình nhận diện nhầm negative thành positive, cần loại bỏ ảnh hưởng của các trường hợp này lên hàm loss, điều chỉnh chúng về 0:

Nếu lựa chọn triplet input ngẫu nhiên thì bất đẳng thức sẽ dễ dàng đạt được vì khả năng để 2 khuôn mặt khác nhau có vector giống nhau rất khó như vậy khiến việc training trở nên thiếu hiệu quả. Do vậy để mô hình có thể phân biệt chuẩn xác hơn giữa các khuôn mặt khác nhau thì cần lựa chọn input khó hơn trong quá trình training. Do đó mỗi anchor cần xác định (hard positive) và (hard negative). Nhóm tác giả bài báo đề xuất hai phương pháp cho vấn đề này:

Tạo triplets offline sau mỗi n bước, sử dụng checkpoint gần nhất và tính argmin, argmax trong bộ dữ liệu con của tập dữ liệu

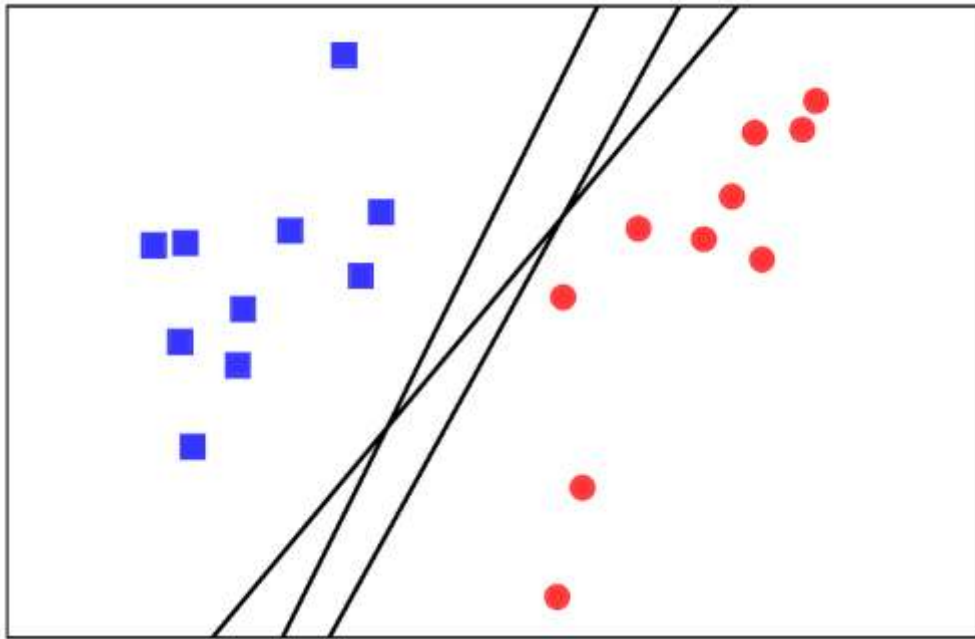
Tạo triplets online. Thực hiện bởi chọn mẫu negative và positive trong mini-batch.

## 6.4. Thuật toán SVM

### 6.4.1. Thuật toán SVM

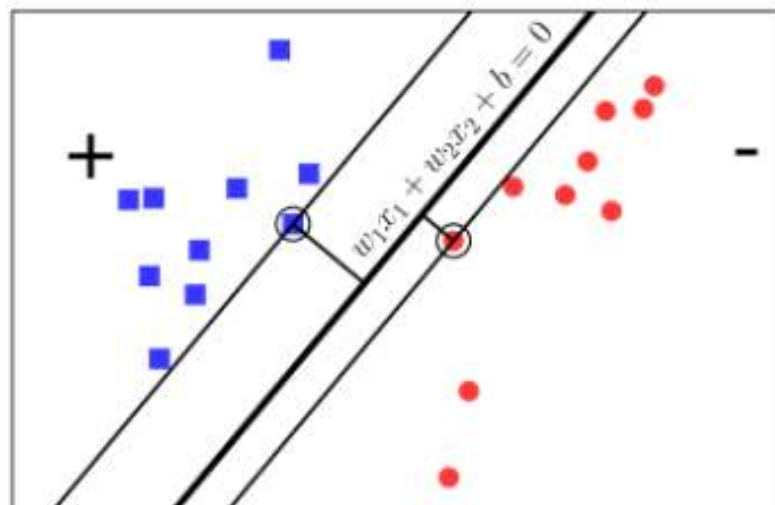
SVM được đề xuất bởi Vladimir N. Vapnik và các đồng nghiệp của ông vào năm 1963 tại Nga và sau đó trở nên phổ biến trong những năm 90 nhờ ứng dụng giải quyết các bài toán phi tuyến tính (nonlinear) bằng phương pháp Kernel Trick.

SVM (Support Vector Machine) là một thuật toán học máy có giám sát được sử dụng rất phổ biến ngày nay trong các bài toán phân lớp (classification) hay hồi quy (Regression). SVM được sử dụng để phân loại dữ liệu bằng cách xác định Hyper plane nhằm phân chia các lớp thành các phần riêng biệt.



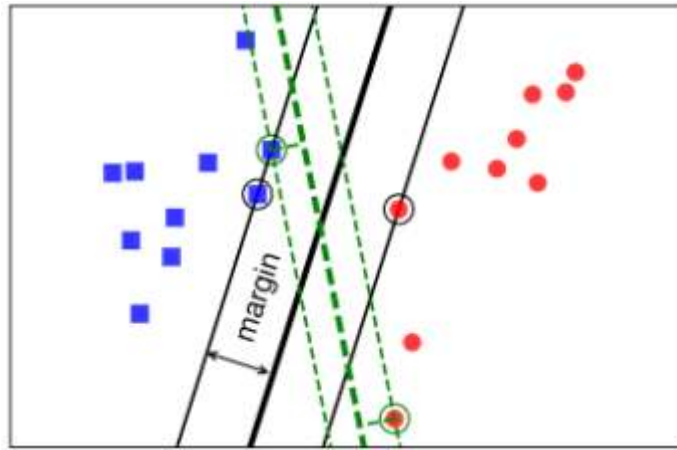
Hình 6. 12. Minh họa phân lớp dữ liệu

Giả sử có hai bộ dữ liệu có sự khác biệt tương đối, có thể xây dựng giữa các bộ dữ liệu một ranh giới phân chia như hình trên. Bài toán đặt ra cần xây dựng một hyper plane tốt nhất theo để có thể dễ dàng phân loại khi có một dữ liệu mới được thêm vào.



Hình 6. 13. Xác định hyper plane

Như hình trên đã xác định hyper plane phân chia 2 bộ dữ liệu tuy nhiên gặp một vấn đề là hyper plane nằm gần bộ dữ liệu đỏ hơn do đó bộ dữ liệu xanh có phần diện tích lớn hơn (Xét trên không gian 2 chiều). Khi đó nếu thêm dữ liệu mới vào sẽ có xu hướng dự đoán về bên xanh nhiều hơn. Do đó để việc dự đoán là tốt nhất cần đảm bảo hyper plane cách đều các bộ dữ liệu nhất có thể. Khoảng cách như nhau này được gọi là margin. (Khi có nhiều bộ dữ liệu trên không gian nhiều chiều sẽ khó đạt được hơn)

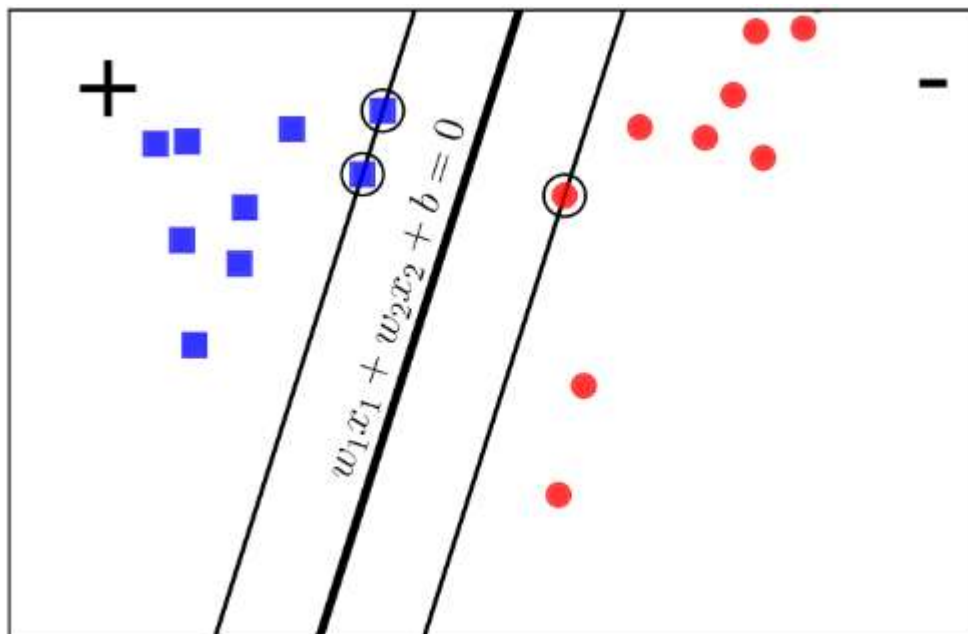


Hình 6. 14. Tối ưu hyper plane

Trên hình tuy đường nét liền màu lục đã đáp ứng được hyper plane cách đều 2 bộ dữ liệu nhưng margin chưa được tối ưu so với đường nét đứt màu đen. Các đường nét đứt màu đen cho margin rộng hơn sẽ mang lại hiệu ứng phân loại tốt hơn vì sự phân chia các bộ dữ liệu rành mạch hơn. Do đó để SVM hoạt động hiệu quả cần tối ưu margin hay tìm hyper plane để margin là lớn nhất.

#### 6.4.2. Tối ưu SVM

Giả sử có bộ dữ liệu training set là với là vector đầu vào của bộ dữ liệu và là nhãn của dữ liệu đó trong không gian  $d$  chiều. Xét bộ dữ liệu chỉ gồm 2 lớp nên nhãn dữ liệu có thể là:



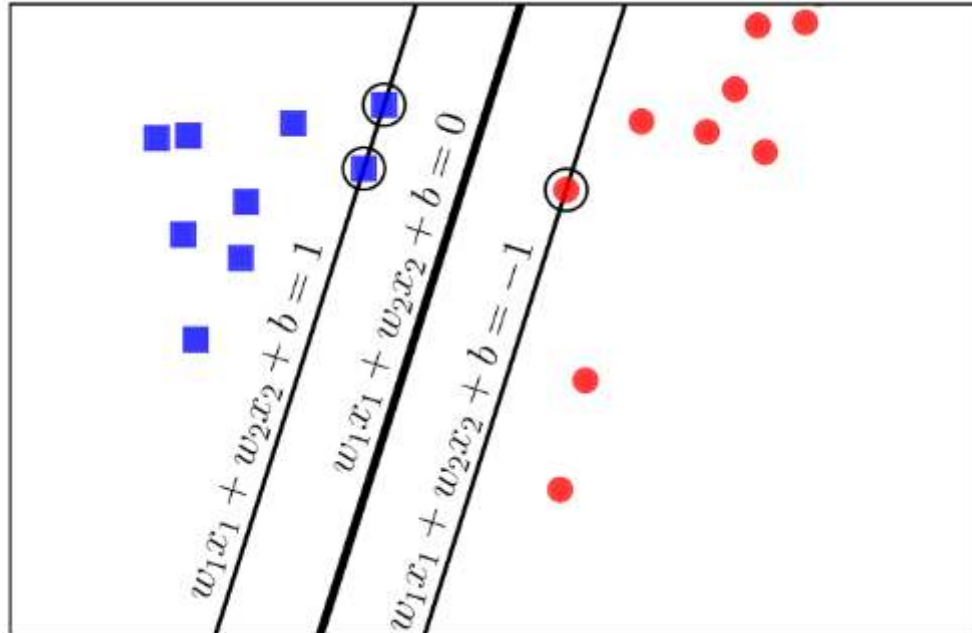
Hình 6. 15. Phân lớp bộ dữ liệu

Có điểm xanh thuộc class 1, điểm đỏ thuộc class -1 và hyper plane

Để xác định được hyper plane ta cần xác định và . Khoảng cách từ điểm dữ liệu bất kì tới hyper plane:

Margin được xác định:

Do đó bài toán tối ưu Margin cần tìm và sao cho margin là lớn nhất:



Hình 6. 16. Tối ưu margin

Xét các điểm xanh

Chuyển về bài toán có ràng buộc:

Lấy nghịch đảo hàm, bình thương, đạo hàm và nhân với

Đến đây bài toán đã có thể xác định được các tham số.

Khi có dữ liệu mới vào bộ dữ liệu chỉ cần truyền vào công thức

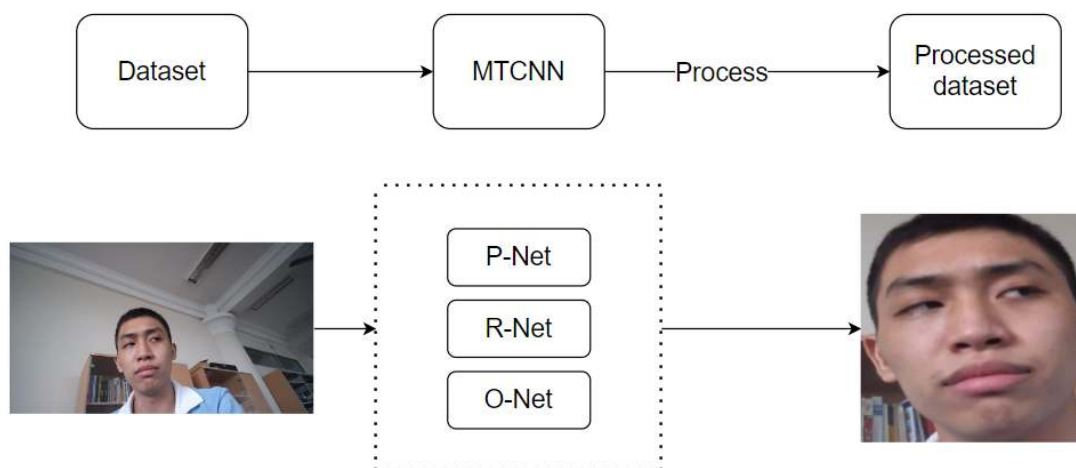
Giá trị trả về sẽ dự đoán lớp của đối tượng mới được thêm vào.

Tương tự ta có thể mở rộng thuật toán trong không gian d chiều để áp dụng trong bài toán.

### 6.5. Ứng dụng hệ thống nhận diện khuôn mặt trong đề tài

Thuật toán MTCNN được áp dụng để nhận diện và định vị khuôn mặt trong khung hình, bộ dữ liệu đầu vào đi qua thuật toán sẽ được tách khuôn mặt có trong khung hình và resize về kích thước chuẩn. Đối với quá trình training, MTCNN giúp tiền xử lý bộ dữ liệu trước khi được đưa vào thuật toán FaceNet, tương tự với quá trình nhận diện, MTCNN cũng xác định tất cả khuôn mặt trên khung hình để đưa vào các thuật toán tiếp theo.

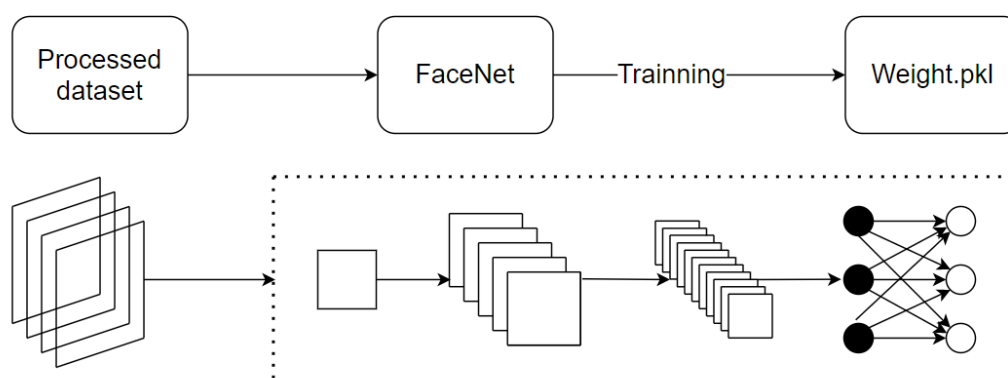




*Hình 6. 17. Tiền xử lý bộ dữ liệu*

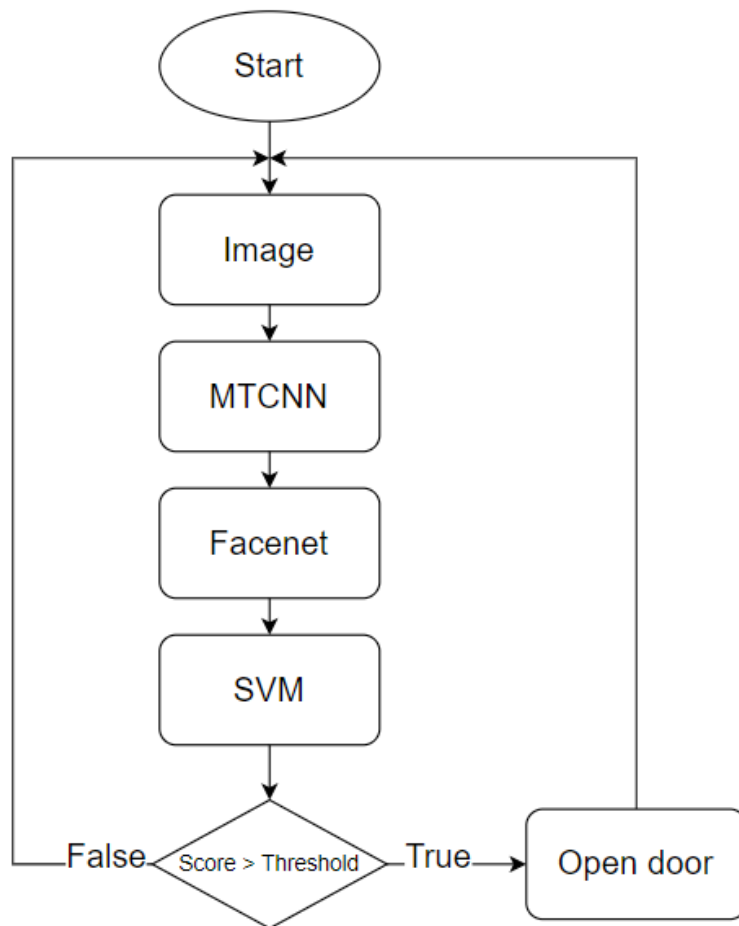
Trong quá trình training: Bộ dữ liệu sau khi tiền xử lý bởi MTCNN được thuật toán FaceNet training và trích xuất các đặc trưng để xuất ra file trọng số (weight.pkl) phục vụ quá trình nhận diện sau này. Để quá trình training diễn ra hiệu quả nhất nhóm sử dụng pre-train weight là 20180402-114759.pb. File trọng số này được training theo kiến trúc Inception ResNetv1 với bộ dữ liệu VGGFace2 với khoảng 3.3 triệu khuôn mặt chia thành 9000 đối tượng, do đó độ chính xác trong quá trình nhận diện đạt 99.65%.

Trong quá trình nhận diện: Sau khi đi qua FaceNet, khuôn mặt được vector hóa thành vector 128 chiều và qua thuật toán SVM để phân lớp từ đó xác định đối tượng.



*Hình 6. 18. Quá trình training*

Dưới đây là sơ đồ quá trình nhận diện.



Hình 6. 19. Quá trình nhận diện

## CHƯƠNG 7. NGHIÊN CỨU PHÁT TRIỂN DESKTOP APP

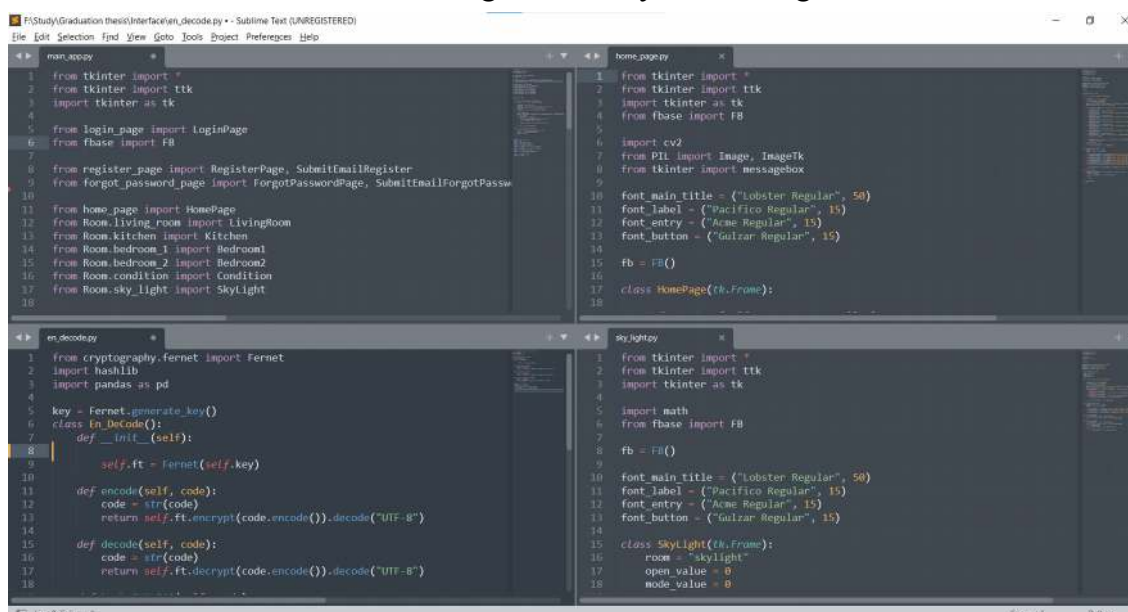
### 7.1. Công cụ và ngôn ngữ lập trình

Phần mềm được viết trên trình soạn thảo Sublime Text 3 – phần mềm lập trình đa nền tảng với nhiều ngôn ngữ trong đó có Python3. App được phát triển trên ngôn ngữ Python 3.7.9 và thư viện Tkinter cung cấp những công cụ mạnh mẽ và hữu ích cho phát triển giao diện qua đó giúp người dùng dễ dàng điều khiển các thiết bị trong nhà với máy tính.

#### 7.1.1. Cấu trúc phần mềm

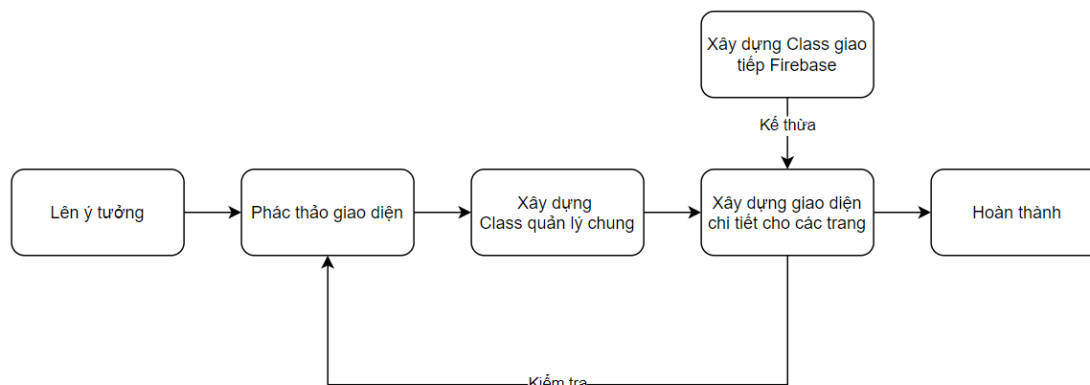
Phần mềm được xây dựng trên các class bao gồm việc điều khiển chung, quản lý trang, các class hỗ trợ

- Class App quản lý trang hiện tại được trình chiếu
- Các Class xây dựng giao diện hiển thị tới người dùng
- Các Class hỗ trợ: mã hóa giải mã, truy xuất thông tin với Firebase, ...



Hình 7. 1. 04 class sử dụng để xây dựng Desktop App

#### 7.1.2. Quy trình xây dựng phần mềm



Hình 7. 2. Sơ đồ quy trình thiết kế phần mềm

Đầu tiên xây dựng class quản lý chung các thành phần của phần mềm là các trang hiển thị và chương trình main. Dựa trên phác thảo và danh sách các chức năng ban đầu sẽ lần lượt xây dựng cụ thể các trang hiển thị. Với mỗi trang sẽ có các chức năng đặc biệt sẽ kế thừa lại các phương thức của các class hỗ trợ. Cuối cùng đối chiếu các chức năng, đề xuất các trường hợp người dùng có thể xảy ra, để xử lý ngoại lệ và điều chỉnh các chức năng phù hợp.

Ưu điểm của việc xây dựng các class (OOP) lưu trong từng file riêng biệt sẽ giúp dễ bảo trì và chỉnh sửa, tối ưu mã nguồn, dễ kế thừa lại các chức năng của class khác.

Thiết kế các thành phần cho giao diện của mỗi trang hiển thị người dung cần định nghĩa tối thiểu 2 thành phần bao gồm: Các thuộc tính hiển thị (Text, màu, font, ...) và vị trí với kích thước, đối với các nút ấn cần định nghĩa hàm chức năng thực hiện.

```
self.living_room_button = Button(self, text='Living Room', font=font_button, relief=SOLID,
                                command=lambda:controller.show_frame("LivingRoom"))
```

Nút ấn chuyển hướng tới trang hiển thị phòng khách thuộc class HomePage, các thuộc tính của nút ấn được định nghĩa như: font, text, relief. Hàm chức năng chuyển hướng tới trang hiển thị được định nghĩa bằng hàm lambda, đối với các nút ấn khác có chức năng phức tạp hơn có thể định nghĩa thành một phương thức của class.

```
self.living_room_button.place(x=framew//2-100, y=180, width=200, height=30)
```

Vị trí của nút ấn được định nghĩa dựa trên chiều dài chiều rộng của phần mềm, sẽ giúp tùy biến khi người dùng thay đổi kích thước phần mềm.

## 7.2. Ứng dụng dịch vụ Firebase trong phần mềm

Class “FB” được xây dựng với phương thức truy xuất dữ liệu từ FireBase và các thuộc tính lưu trữ key để truy cập tới project trên FireBase.

Đẩy dữ liệu từ phần mềm lên FireBase

```
def push_data(self, room, obj, value):
    self.db.child(room).update({obj: value})
```

Đọc dữ liệu từ FireBase xuống phần mềm

```
def get_data(self, room, obj):
    return self.db.child(f"{room}/{obj}").get().val()
```

## 7.3. Xây dựng giao diện Desktop App

Giao diện sẽ được chia thành các file có extension là “.py”. Trước tiên tiến hành xây dựng class App trong file main\_app.py để quản lý các trang hiển thị và phương thức hiển thị trang khi có yêu cầu.

```

class App(tk.Tk):
    def __init__(self, *args, **kwargs):
        tk.Tk.__init__(self, *args, **kwargs)

        container = tk.Frame(self)
        container.pack(side="top", fill="both", expand=True)
        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)

        self.frames = {}
        pages = (LoginPage, RegisterPage, ForgotPasswordPage, SubmitEmailRegister, SubmitEmailForgotPassword,
                  HomePage, LivingRoom, Kitchen, Bedroom1, Bedroom2, Condition, SkyLight, ChangePasswordPage)
        for F in pages:
            page_name = F.__name__
            frame = F(parent=container, controller=self)
            self.frames[page_name] = frame

            frame.grid(row=0, column=0, sticky="nsew")

        self.show_frame("LoginPage")

    def show_frame(self, page_name):
        '''Show a frame for the given page name'''
        frame = self.frames[page_name]
        frame.tkraise()

```

Sau đó tiến hành xây dựng chi tiết các trang hiển thị, tại các trang sẽ có nút chuyển tiếp tới các trang khác, khi đó controller sẽ quyết định chuyển tiếp tới trang hiện thị được gọi. Các class trang hiển thị sẽ bao gồm các: Button, Label, Entry, ...

Các class được xây dựng tương tự nhau về mặt cấu trúc

```

class HomePage(tk.Frame):
    def __init__(self, parent, controller):
        tk.Frame.__init__(self, parent)
        self.controller = controller
        self.configure(bg="#01C5FE")
        label = tk.Label(self, text="QASHome", bg="#01C5FE", font=font_main_title)
        label.pack(side="top", fill="x", pady=10)

        self.living_room_button = Button(self, text='Living Room', font=font_button, relief=SOLID,
                                          command=lambda:controller.show_frame("LivingRoom"))
        self.kitchen_button = Button(self, text='Kitchen', font=font_button, relief=SOLID,
                                      command=lambda:controller.show_frame("Kitchen"))
        self.bedroom_1_button = Button(self, text='Bedroom 1', font=font_button, relief=SOLID,
                                       command=lambda:controller.show_frame("Bedroom1"))
        self.bedroom_2_button = Button(self, text='Bedroom 2', font=font_button, relief=SOLID,
                                       command=lambda:controller.show_frame("Bedroom2"))
        self.bathroom_button = Button(self, text='Life condition', font=font_button, relief=SOLID,
                                      command=lambda:controller.show_frame("Condition"))
        self.sky_light_button = Button(self, text='Sky light', font=font_button, relief=SOLID,
                                       command=lambda:controller.show_frame("SkyLight"))
        self.voice_button = Button(self, text="Voice", font=font_button, relief=SOLID,
                                   command=self.voice)

        self.logout_button = Button(self, text="Logout", font=font_button, relief=SOLID,
                                    command=self.logout)
        self.update()
        # Put them on the frame
        self.bind("<Configure>", self.placeGUI)

    def placeGUI(self, e):
        frameW = self.winfo_width()
        frameH = self.winfo_height()

        self.living_room_button.place(x=frameW//2-100, y=180, width=200, height=30)

```

Đoạn mã trên thực hiện xây dựng trang hiển thị HomePage. Mỗi trang hiển thị gồm các phần: thuộc tính trang, hàm khai báo các thành phần của trang, hàm định vị tọa độ của các thành phần và các hàm có chức chuyên biệt của trang.



## CHƯƠNG 8. NGHIÊN CỨU PHÁT TRIỂN APP ANDROID

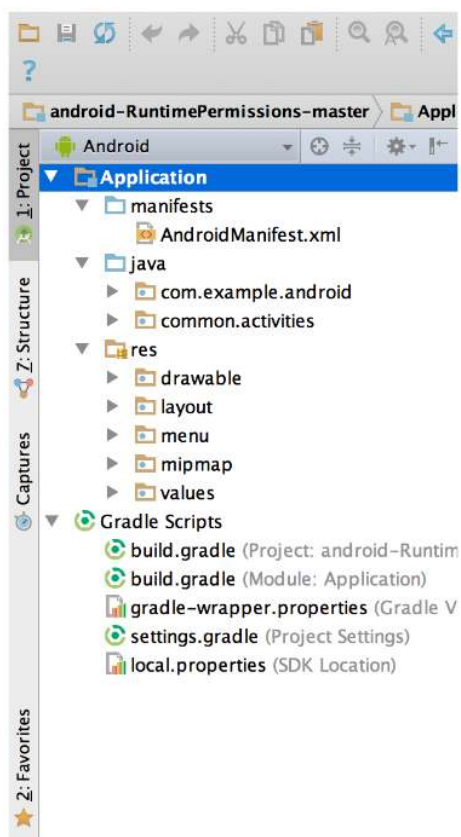
### 8.1. Công cụ và ngôn ngữ lập trình

Android Studio là môi trường phát triển tích hợp (IDE) chính thức để phát triển ứng dụng Android, dựa trên IntelliJ IDEA. Dựa trên các trình soạn thảo mã và công cụ phát triển mạnh mẽ của IntelliJ, Android còn cung cấp thêm nhiều tính năng giúp người dùng nâng cao năng suất khi xây dựng ứng dụng Android, chẳng hạn như: mô phỏng nhanh và nhiều tính năng, môi trường hợp nhất, đa dạng khung và công cụ thử nghiệm...

#### 8.1.1. Cấu trúc dự án

Mỗi dự án trong Android Studio chứa một hoặc nhiều module có tệp mã nguồn và tệp tài nguyên, bao gồm các loại module sau:

- Module ứng dụng Android
- Module thư viện
- Module google App Engine



Hình 8. 1. Các tệp dự án trong chế độ xem “Android”

Theo mặc định, Android Studio thể hiện các tệp dự án trong chế độ xem “Android” như trên hình. Chế độ xem này được sắp xếp theo module để người dùng có thể truy cập nhanh chóng vào các tệp nguồn chính của dự án. Trong mỗi module ứng dụng chứa các thư mục sau:

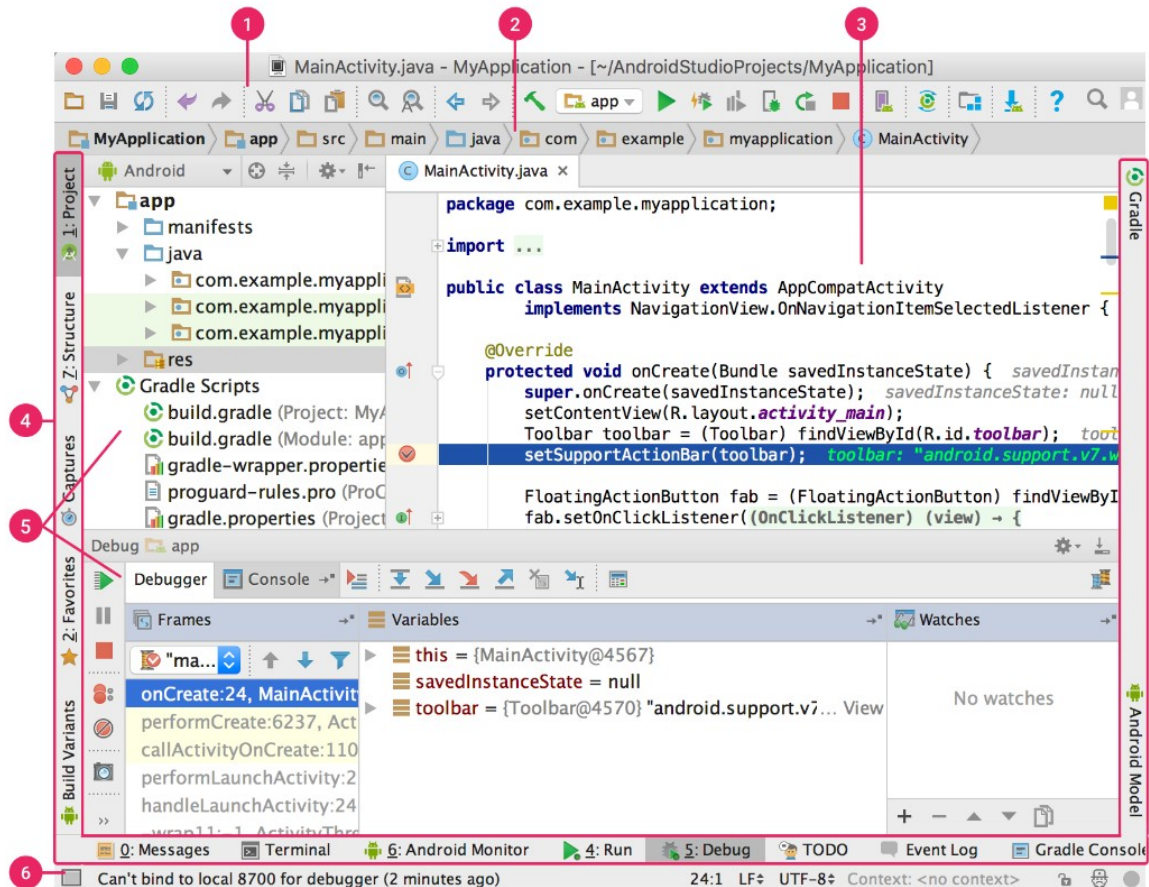
- **manifests**: Chứa tệp AndroidManifest.xml
- **java**: Chứa các tệp mã nguồn Java, bao gồm cả mã kiểm thử Junit



- **res**: Chứa mọi tài nguyên không ở dạng mã, chẳng hạn như bố cục XML, chuỗi giao diện người dùng, hình ảnh sử dụng trong XML...

### 8.1.2. Giao diện người dùng trên Android Studio

Cửa sổ chính của Android Studio bao gồm các khu vực logic:

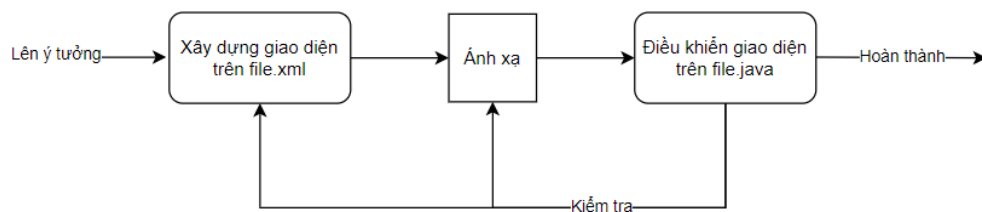


Hình 8. 2. Giao diện người dùng trên Android Studio

- **Toolbar** (Thanh công cụ): Cho phép thực hiện nhiều thao tác, bao gồm cả việc chạy ứng dụng Android và chạy các công cụ Android
- **Navigation bar** (Thanh điều hướng): Giúp người dùng điều hướng trong dự án và mở các tệp chỉnh sửa
- **Editor window** (Cửa sổ trình chỉnh sửa): Là nơi tạo và sửa đổi code
- **Tool window bar** (Thanh cửa sổ công cụ): Nằm xung quanh, bên ngoài cửa sổ IDE và chứa các nút cho phép người dùng mở rộng hoặc thu gọn từng cửa sổ công cụ riêng lẻ
- **Tool windows** (Cửa sổ công cụ): Cho phép truy cập vào các công việc cụ thể như quản lý dự án, tìm kiếm, kiểm soát phiên bản...
- **Status bar** (Thanh trạng thái): Cho thấy trạng thái của dự án cũng như mọi thông báo, cảnh báo

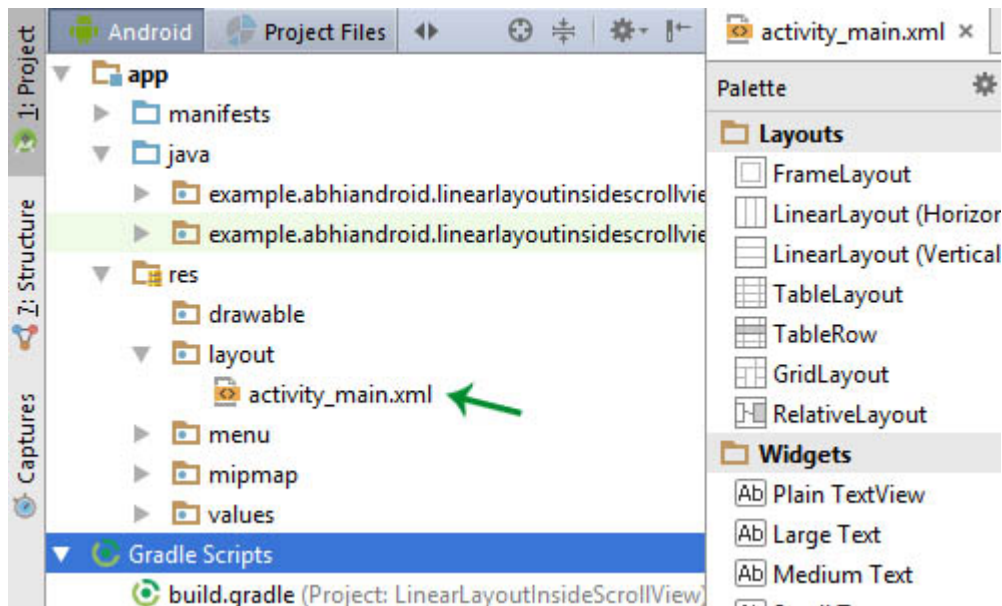


### 8.1.3. Quy trình các bước lập trình trên Android Studio



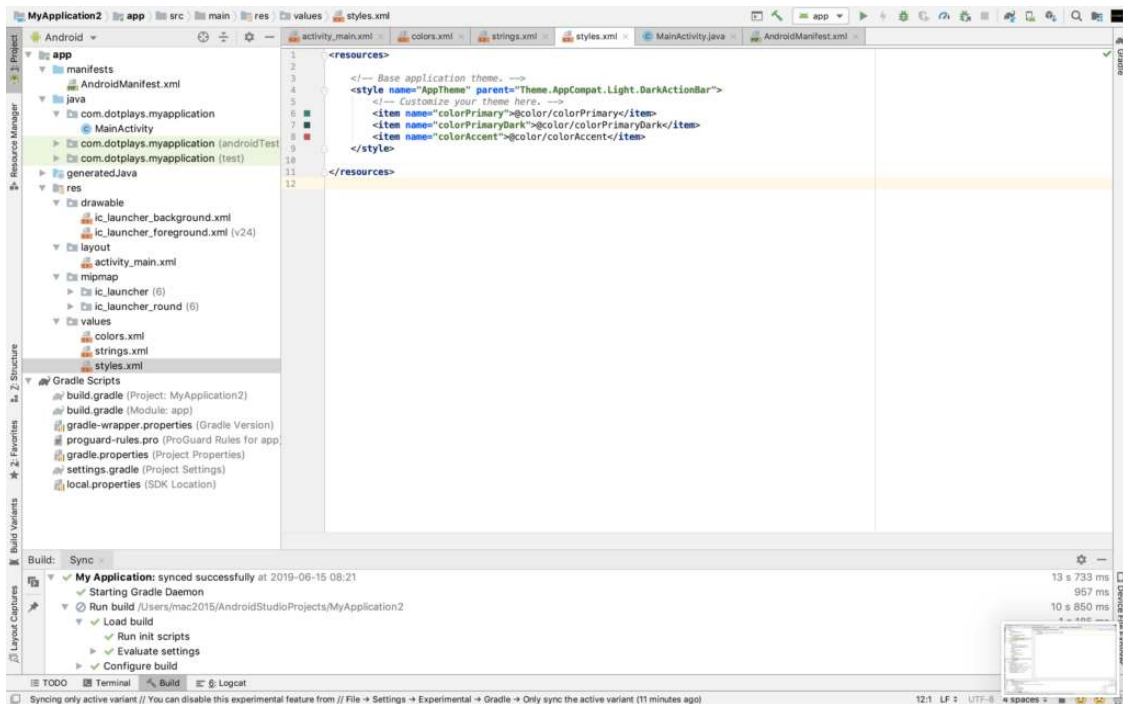
Hình 8. 3. Quy trình xây dựng giao diện Android App

Xây dựng giao diện trên file.xml chính là thiết kế giao diện phía trước (hay còn gọi là Front End) hiển thị trên ứng dụng của người dùng.



Hình 8. 4. File.xml trong tool windows

Người lập trình có thể tạo mới các file.xml khác có đuôi .xml với lưu ý: mỗi file.xml biểu diễn một màn hình hiển thị trên ứng dụng. Sau khi lựa chọn file.xml để thiết kế giao diện, bên trái sẽ xuất hiện giao diện:



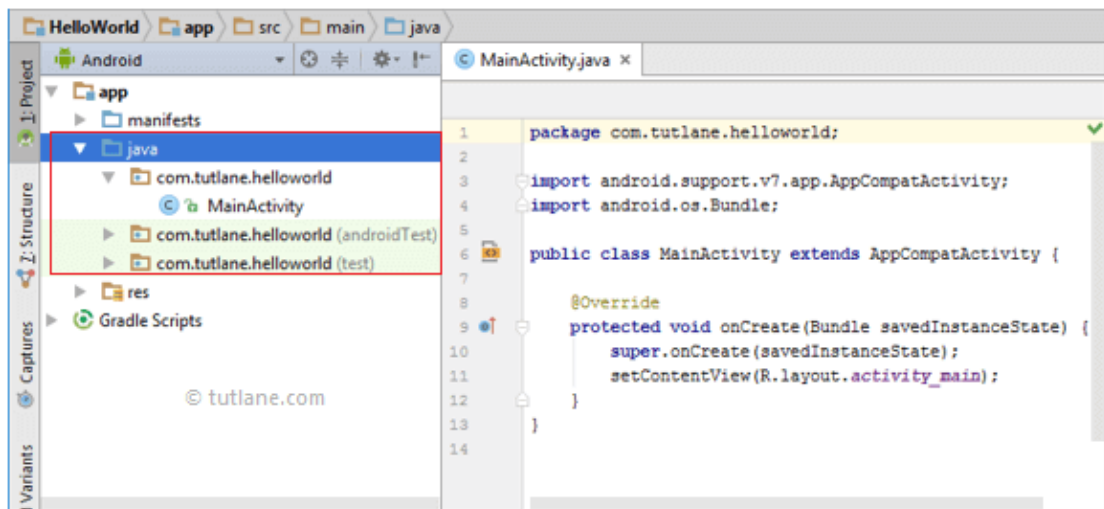
Hình 8. 5. Giao diện file.xml

Thiết kế giao diện cho ứng dụng có hai cách, người dùng có thể lựa chọn sử dụng một trong hai cách hoặc đồng thời cả hai bao gồm: hình thức viết code và hình thức kéo thả các View vào màn hình ảo.

Quá trình ánh xạ có thể hiểu đơn giản là bước tiền xử lý cho quá trình điều khiển giao diện trên file.java. Sau khi thiết kế giao diện phần nhìn cho ứng dụng, người phát triển cần đặt các định danh id cho các View trong giao diện, từ các id đó, người phát triển sẽ thực hiện quá trình “Ánh xạ” tới file.java, tức là các đối tượng được lập trình trong file.java có thể hiểu rằng mình đang làm việc với View nào trong file.xml. Một ví dụ về quá trình “Ánh xạ” như sau:

```
Button btnDN = (Button) findViewById(R.id.btnDangNhap);
```

Điều khiển giao diện trên file.java là quá trình lập trình trên file.java nhằm điều khiển được những trường hợp người dùng thực hiện trên giao diện đã được xây dựng trước đó (như: bấm, kéo thả, giữ...). Trên Android Studio, mỗi một file.java sẽ tương thích với chỉ một file.xml và trên file.java sẽ có các đoạn lệnh điều khiển các View trên file.xml tương ứng.



Hình 8. 6. File.java trên Android Studio

## 8.2. Ứng dụng dịch vụ Firebase trong Android Studio

Như đã giới thiệu ở chương 3, dịch vụ Firebase Realtime Database chính là khu vực lưu trữ thông tin điều khiển nhà thông minh thông qua các tác động từ App Android. Tại các file.java, dữ liệu được đẩy lên hoặc thu về từ Firebase Realtime Database bằng hai phương thức như sau:

- Đọc dữ liệu từ Firebase xuống Android:

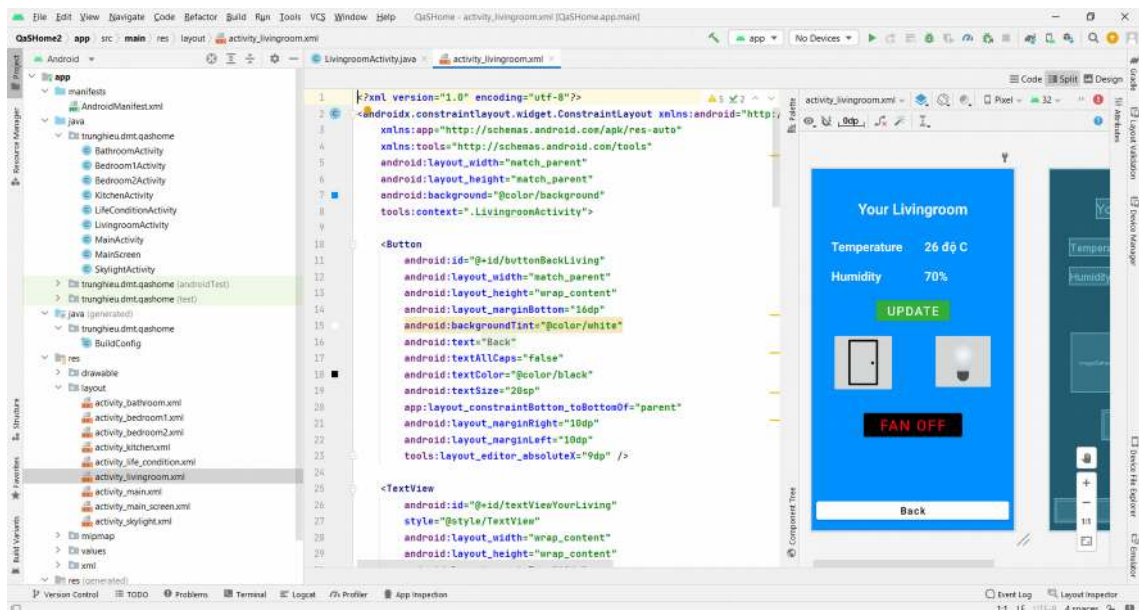
```
// Đọc dữ liệu temp
tempDatabase = FirebaseDatabase.getInstance().getReference().child("/livingroom/temp");
tempDatabase.addValueEventListener(new ValueEventListener() {
```

- Đẩy dữ liệu từ Android lên Firebase:

```
// Ghi dữ liệu đóng cửa
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference( path: "/livingroom/door");
myRef.setValue("0");
```

## 8.3. Xây dựng giao diện Android App

Giao diện sẽ được viết trong file activity\_livingroom.xml, Android Studio có hỗ trợ các View để tạo giao diện với chức năng tương ứng như: Button, TextView, Slider... Sau khi xây dựng xong một View trong giao diện, cần phải gán cho View đó một ID xác thực, duy nhất.



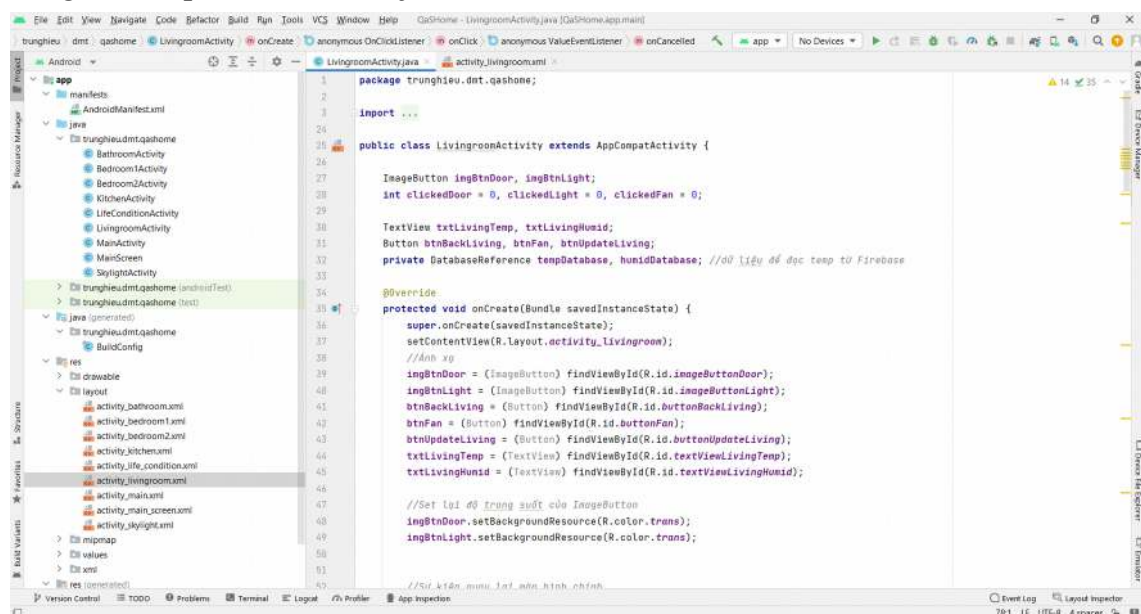
Hình 8. 7. Ví dụ về xây dựng giao diện phòng khách

Để có thể điều khiển các View được lựa chọn trên file.xml, ta cần ánh xạ các View đó tới file.java như ở bước 2 được giới thiệu trong mục 8.1.3.

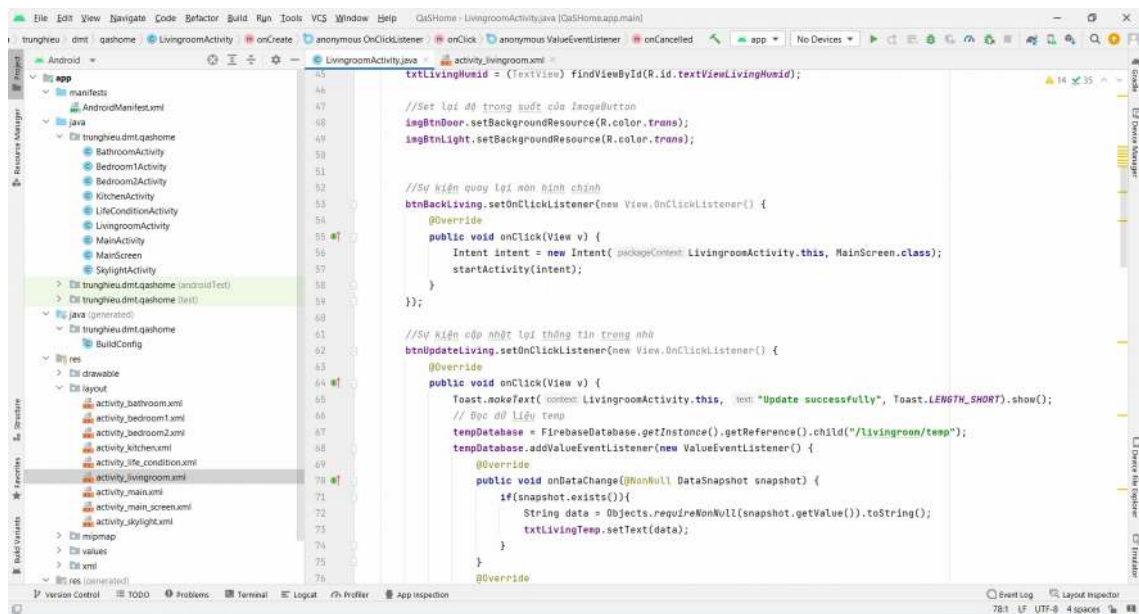
//Ánh xạ

```
imgBtnDoor = (ImageButton) findViewById(R.id.imageButtonDoor);
imgBtnLight = (ImageButton) findViewById(R.id.imageButtonLight);
btnBackLiving = (Button) findViewById(R.id.buttonBackLiving);
btnFan = (Button) findViewById(R.id.buttonFan);
btnUpdateLiving = (Button) findViewById(R.id.buttonUpdateLiving);
txtLivingTemp = (TextView) findViewById(R.id.textViewLivingTemp);
txtLivingHumid = (TextView) findViewById(R.id.textViewLivingHumid);
```

Sau khi ánh xạ, ta có thể thực hiện điều khiển cả trên giao diện và trên Firebase bằng cách lập trình trên file.java







*Hình 8. 8. Ví dụ xây dựng giao diện phòng khách trên file.java*

Các giao diện khác thực hiện tương tự như trên, phần source code sẽ được gửi kèm trong phần phụ lục của đồ án.

## CHƯƠNG 9. KẾT QUẢ

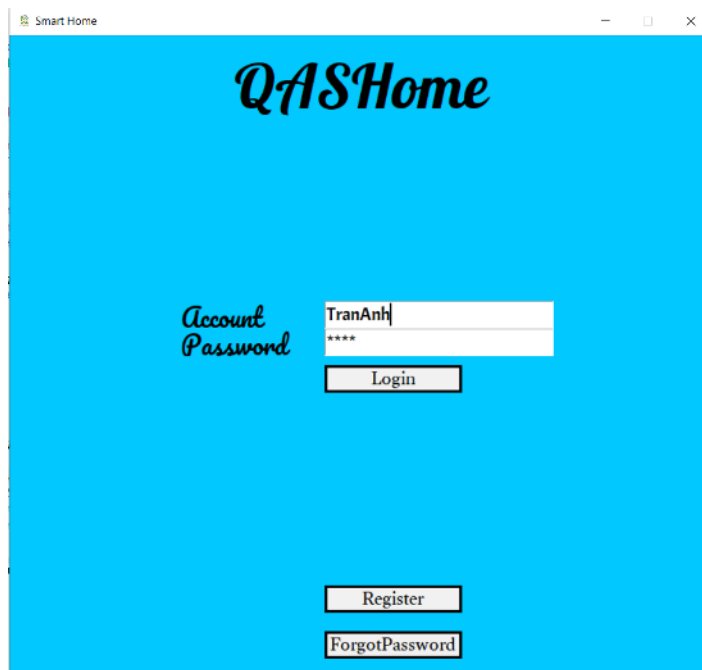
### 9.1. Mô hình nhà thông minh



*Hình 9. 1. Mô hình nhà thông minh thực tế*

### 9.2. Giao diện Desktop App

Người dùng cần thực hiện đăng nhập với tài khoản mật khẩu đã đăng ký trước, chỉ khi đăng nhập thành công người dùng mới truy cập được vào trang chính. Khi người dùng ấn nút “Login” phần mềm sẽ tiến hành đọc account và password và sử dụng mã hóa SHA-512 để tìm và đối chiếu với account và password trong cơ sở dữ liệu.



*Hình 9. 2. Giao diện đăng nhập của ứng dụng Desktop App*

Nếu người dùng chưa có tài khoản có thể bấm nút “Register” để tiến hành đăng ký tài khoản. Tại đây người dùng được yêu cầu điền đầy đủ thông tin theo mẫu. Hệ thống sẽ ghi nhận thông tin và gửi mã code xác nhận tới người dùng thông qua email vừa đăng ký, nếu người dùng nhập code thành công hệ thống sẽ ghi thông tin vào cơ sở dữ liệu. Tại đây account và password sẽ được mã hóa SHA-512 do các thông tin này không cần giải mã ngược để sử dụng, chỉ cần đối chiếu so sánh với thông tin mới sau khi mã hóa nhằm tăng tính bảo mật cho tài khoản người dùng. Đối với các thông tin còn lại sẽ tiến hành encryption do các thông tin này cần decryption để sử dụng một cách rõ ràng trong các trường hợp cần thiết.

Hình 9. 3. Giao diện đăng ký của ứng dụng

Tại giao diện “Forgot Password” người dùng có thể tiến hành thay đổi tìm lại tài khoản bằng cách nhập account và email, nếu thông tin chính xác hệ thống sẽ gửi mã code tới người dùng thông qua email. Người dùng sau khi nhập mã xác minh sẽ được chuyển tới giao diện đổi mật khẩu. Hệ thống sẽ ghi nhận mật khẩu mới của người dùng và người dùng có thể sử dụng mật khẩu mới để đăng nhập.

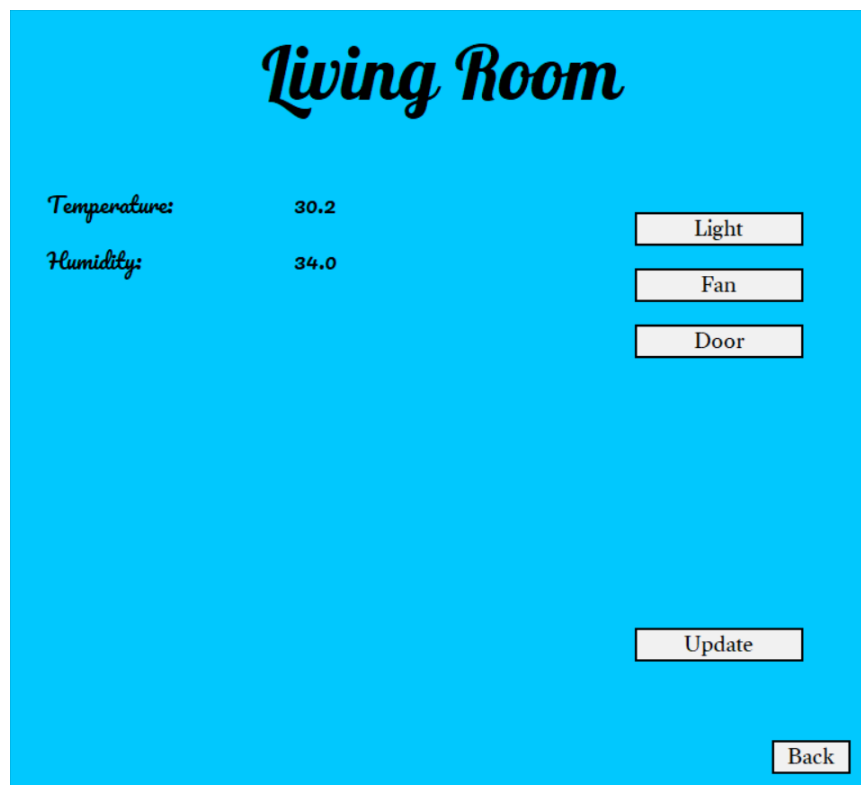
Hình 9. 4. Nếu người dùng quên tài khoản mật khẩu

Tại giao diện chính của phần mềm người dùng có thể tùy chọn tới các chức năng của phần mềm hoặc đăng xuất. Khi đăng xuất hệ thống sẽ gửi cảnh báo trước khi người dùng đăng xuất khỏi ứng dụng.



Hình 9. 5. Giao diện "Home" của Desktop App

Tại giao diện quản lý của ứng dụng có các chức năng: Cập nhật lại nhiệt độ và độ ẩm của phòng khách tại nút "Update". Các nút "Light", "Fan", "Door" giúp người dùng tương tác trực tiếp với các thiết bị trong phòng. Người dùng có thể lựa chọn "Back" để quay trở lại trang chủ của ứng dụng.



Hình 9. 6. Giao diện quản lý Living room của Desktop App



Tại giao diện quản lý của ứng dụng có các chức năng: Cập nhật lại nhiệt độ và độ ẩm của phòng bếp tại nút “Update”. Các nút “Light”, “Curtain” giúp người dùng tương tác trực tiếp với các thiết bị trong phòng. Người dùng có thể lựa chọn “Back” để quay trở lại trang chủ của ứng dụng.



Hình 9. 7. Giao diện quản lý Kitchen của Desktop App



Hình 9. 8. Giao diện quản lý Bedroom 1 của Desktop App



Hình 9. 9. Giao diện quản lý Bedroom 2 của Desktop App



Hình 9. 10. Giao diện quản lý khu vực giếng trời của Desktop App

Giao diện “Life condition” cung cấp thông tin về môi trường xung quanh như: tốc độ gió, độ ẩm, thời tiết hiện tại, ... Khi người dùng ấn nút “Update” các dữ

liệu sẽ được cập nhật lại. Các thông tin trên được lấy xuống từ FireBase do phần mềm quản lý thời tiết cung cấp.

The screenshot shows a mobile application interface titled "life Condition" in a stylized font. Below the title, there is a list of weather-related data points, each with a label and a value:

- Description: Overcast Clouds
- Location: Hanoi
- Humidity: 73
- Wind speed: 1.53
- Max temperature: 30
- Min temperature: 30
- Temperature: 30
- Temperature feel like: 35.78
- Sunrise: 05:26:16
- Sunset: 18:39:30

At the bottom right of the data list, there is an "Update" button. At the very bottom right of the screen, there is a "Back" button.

Hình 9. 11. Giao diện quản lý điều kiện sống Life Condition

### 9.3. Giao diện Android App

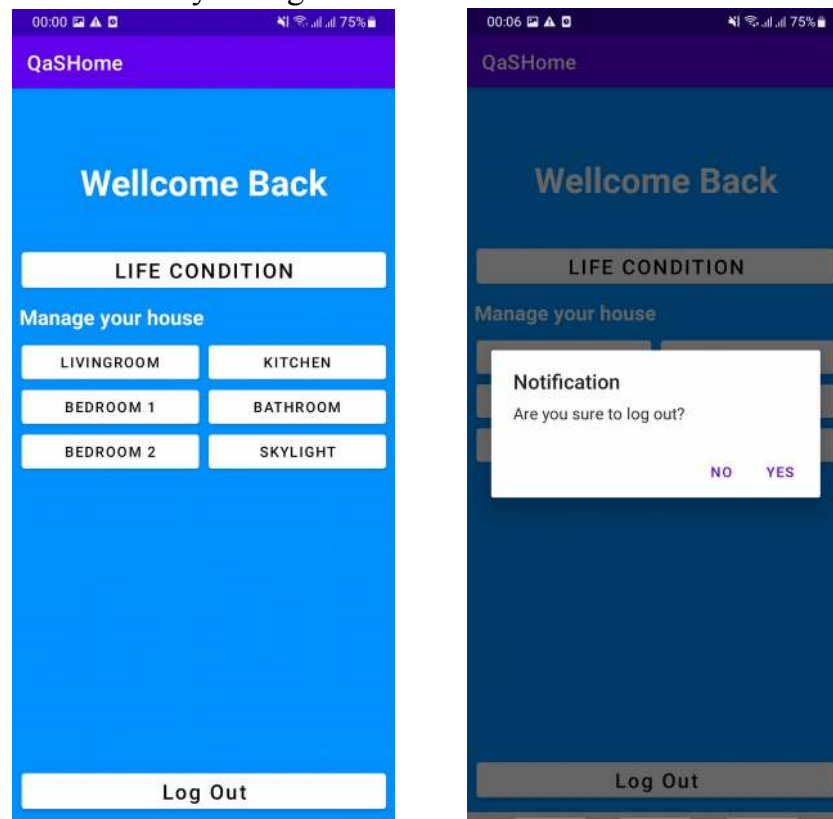
Để tăng tính bảo mật cho ứng dụng, ứng dụng yêu cầu người dùng phải thực hiện đăng nhập bằng tài khoản và mật khẩu của người dùng. Nếu thực hiện sai sẽ yêu cầu người dùng nhập lại tới khi chính xác.

The screenshot shows the login screen of an application named "SQAHome". The screen has a blue header with the app name "SQAHome" in white. Below the header, there are two input fields: "Enter your User Name" and "Enter your Password". Below these fields is a blue button labeled "LOG IN". Below the "LOG IN" button is a link labeled "Forgot Password?". At the bottom of the screen, there is a link labeled "sign up for Smart Home App".

Hình 9. 12. Giao diện đăng nhập của ứng dụng

Giao diện chính của ứng dụng bao gồm các vùng: Vùng đăng xuất, vùng thông báo điều kiện thời tiết trong khu vực và vùng quản lý các phòng trong nhà.

Đối với vùng đăng xuất, người dùng nếu có nhu cầu thoát đăng nhập khỏi thiết bị sẽ nhấn chọn “Log Out”, khi đó sẽ xuất hiện hộp thoại hỏi lại người dùng có chắc chắn muốn thoát ra hay không.



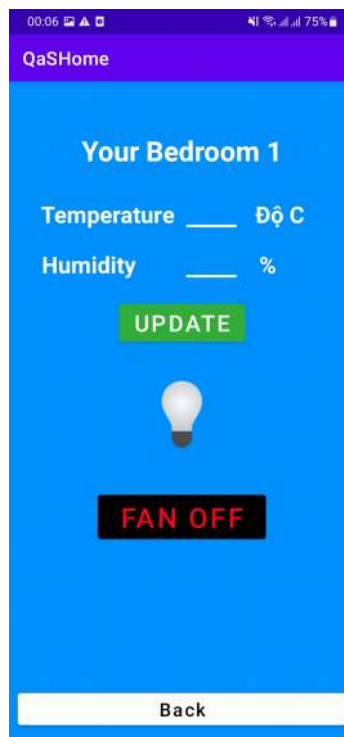
Hình 9. 13. Giao diện chính của ứng dụng

Giao diện quản lý phòng khách của ứng dụng có khả năng: Cập nhật lại nhiệt độ, độ ẩm trong phòng bằng cách nhấn chọn vào button “Update”, khi đó, nhiệt độ, độ ẩm trong phòng sẽ được cập nhật lại trên ứng dụng. Ngoài ra, ứng dụng còn cho phép người dùng đóng mở các thiết bị trong phòng bao gồm: cửa, đèn và quạt bằng cách nhấn chọn vào các icon được hiển thị. Nếu người dùng muốn thoát khỏi trang quản lý phòng khách để tới trang quản lý chung có thể lựa chọn button “Back” để trở lại trang chủ của ứng dụng.



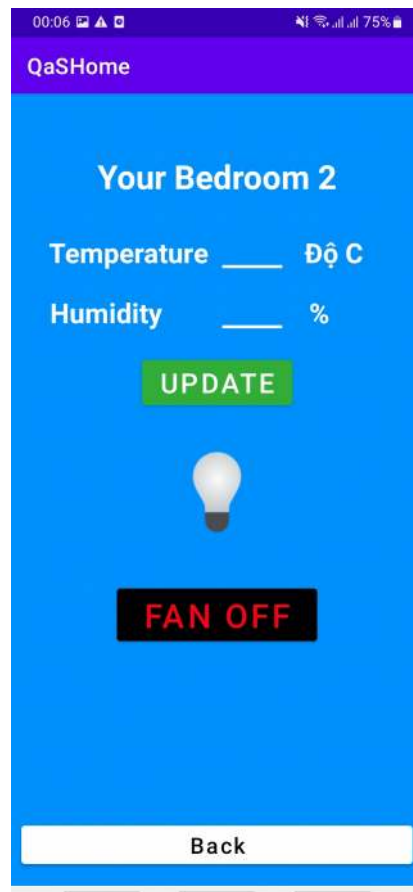
Hình 9. 14. Giao diện quản lý phòng khách

Giao diện quản lý phòng ngủ 1 của ứng dụng có khả năng: Cập nhật lại nhiệt độ, độ ẩm trong phòng bằng cách nhấn chọn vào button “Update”, khi đó, nhiệt độ, độ ẩm trong phòng sẽ được cập nhật lại trên ứng dụng. Ngoài ra, ứng dụng còn cho phép người dùng đóng mở các thiết bị trong phòng bao gồm: đèn và quạt bằng cách nhấn chọn vào các icon được hiển thị. Nếu người dùng muốn thoát khỏi trang quản lý phòng khách để tới trang quản lý chung có thể lựa chọn button “Back” để trở lại trang chủ của ứng dụng.



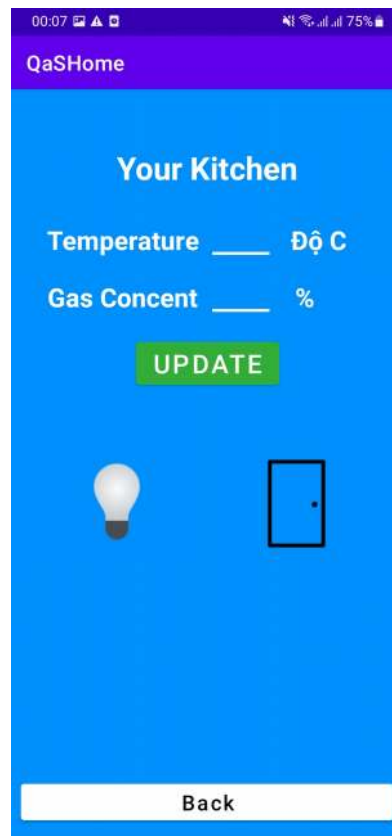
Hình 9. 15. Giao diện quản lý phòng ngủ 1

Giao diện quản lý phòng ngủ 2 của ứng dụng có khả năng: Cập nhật lại nhiệt độ, độ ẩm trong phòng bằng cách nhấn chọn vào button “Update”, khi đó, nhiệt độ, độ ẩm trong phòng sẽ được cập nhật lại trên ứng dụng. Ngoài ra, ứng dụng còn cho phép người dùng đóng mở các thiết bị trong phòng bao gồm: đèn và quạt bằng cách nhấn chọn vào các icon được hiển thị. Nếu người dùng muốn thoát khỏi trang quản lý phòng khách để tới trang quản lý chung có thể lựa chọn button “Back” để trở lại trang chủ của ứng dụng.



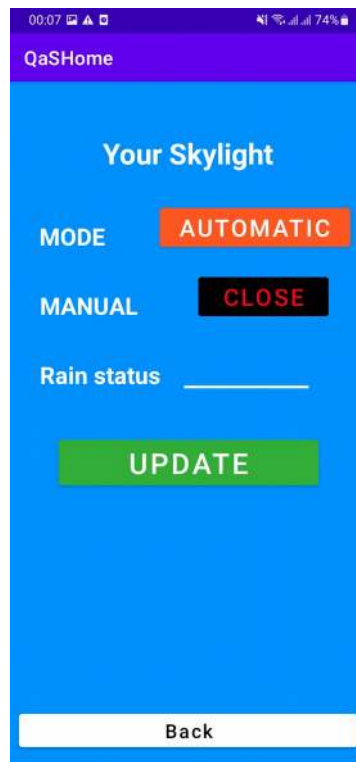
*Hình 9. 16. Giao diện quản lý phòng ngủ 2*

Giao diện quản lý phòng khách của ứng dụng có khả năng: Cập nhật lại nhiệt độ, nồng độ khí gas trong phòng bằng cách nhấn chọn vào button “Update”, khi đó, nhiệt độ, nồng độ khí gas trong phòng sẽ được cập nhật lại trên ứng dụng. Ngoài ra, ứng dụng còn cho phép người dùng đóng mở các thiết bị trong phòng bao gồm: đèn và cửa bằng cách nhấn chọn vào các icon được hiển thị. Nếu người dùng muốn thoát khỏi trang quản lý phòng khách để tới trang quản lý chung có thể lựa chọn button “Back” để trở lại trang chủ của ứng dụng.



*Hình 9. 17. Giao diện quản lý phòng bếp*

Giao diện quản lý khu vực giếng trời của ứng dụng có khả năng: Cập nhật lại tình trạng mưa ở khu vực giếng trời bằng cách nhấn chọn vào button “Update”, khi đó, tình trạng mưa (Có mưa hoặc không mưa) trong khu vực đặt giếng trời sẽ được cập nhật lại trên ứng dụng. Ngoài ra, ứng dụng còn cho phép người dùng lựa chọn 2 chế độ đóng mở cửa giếng trời gồm: manual và automatic. Khi ở chế độ manual, người dùng có thể lựa chọn đóng và mở trực tiếp trên app. Khi ở chế độ automatic, người dùng không thể lựa chọn trên app android mà khi đó, cửa giếng trời sẽ tự động đóng mở tùy thuộc vào tình trạng mưa của khu vực.



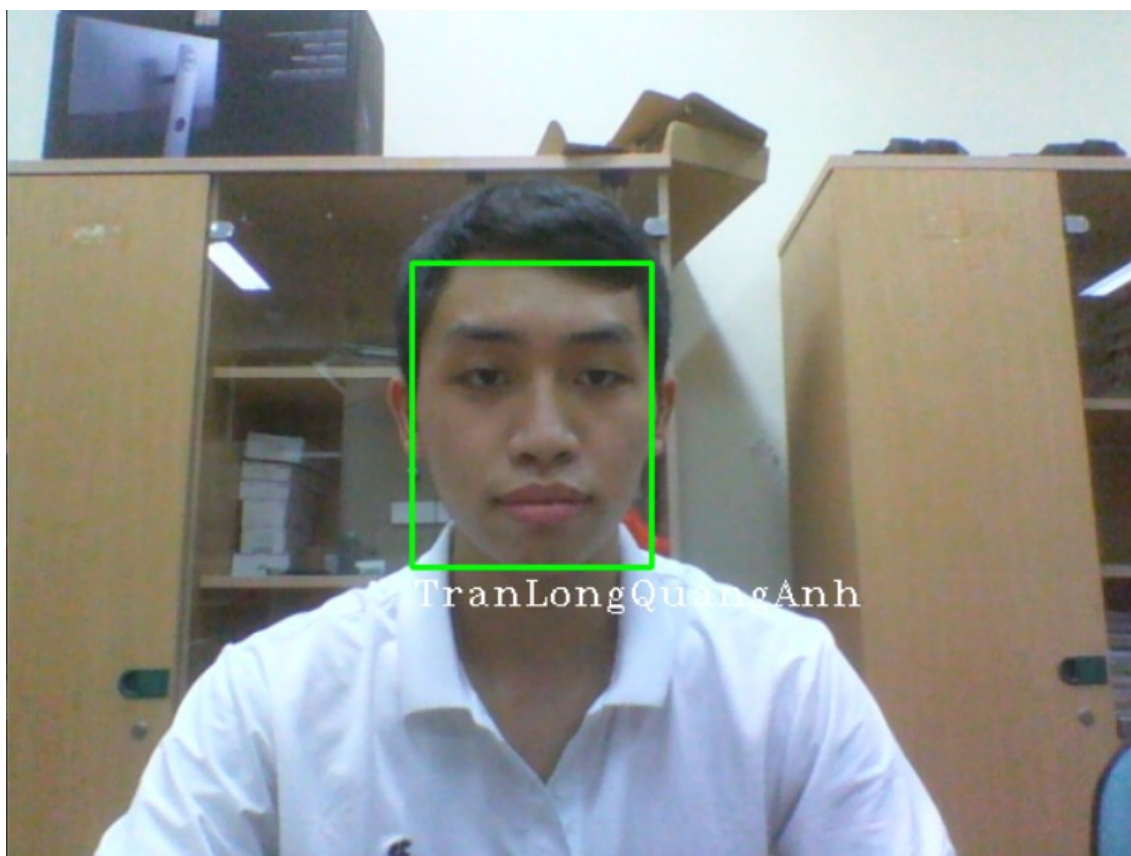
Hình 9. 18. Giao diện quản lý khu vực giếng trời

#### 9.4. Hệ thống nhận diện khuôn mặt



Hình 9. 19. Phân biệt nhiều người trong cùng khung hình





*Hình 9. 20. Nhận diện chính xác người dùng*

## **9.5. Mô hình giếng trời**

## CHƯƠNG 10. KẾT LUẬN

### 10.1. Kết luận

Qua quá trình tìm hiểu lý thuyết, nghiên cứu và tiến hành thực hiện đề tài đồ án tốt nghiệp, nhóm chúng em đã tiếp thu được nhiều kiến thức bổ ích, kinh nghiệm thực hành hơn. Về cơ bản, nhóm chúng em đã hoàn thành sơ bộ đề tài nghiên cứu với chức năng điều khiển bật tắt thiết bị từ xa bằng điện thoại kết hợp phần mềm.

#### a) Về mặt phần cứng

- Sản phẩm chạy ổn định, linh kiện không bị nóng nếu phải hoạt động lâu
- Mô hình tương đối giống thật với các khu vực phòng được bố trí hợp lý

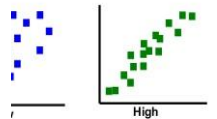
#### b) Về mặt phần mềm

- Thiết kế app có giao diện thân thiện với người dùng, dễ sử dụng, dễ mở rộng khi kết nối với thiết bị khác
- App Android chạy ổn định, quá trình liên kết với Firebase không ngắt quãng nếu hoạt động trong môi trường có mạng Wifi, 4G...

### 10.2. Hướng phát triển của đồ án trong tương lai

Nhận thấy một số bất cập trong khi thực hiện đề tài, nhóm chúng em xin đề xuất một số hướng phát triển trong tương lai nhằm phù hợp hoá nội dung của đề tài:

- Cải thiện thêm chức năng của phần cứng, cho phép mở rộng thêm với các loại cảm biến khác như cảm biến ánh sáng, cảm biến chuyển động...
- Xem xét tới việc sử dụng các loại cảm biến cao cấp được sử dụng trong các mô hình nhà thông minh thực.
- Cho phép ghép nối mạch hiện tại với các module mạch khác.
- Cải thiện thêm về mặt ứng dụng người dùng, thêm vào đó một số tính năng mới như vẽ biểu đồ tham số môi trường, lịch sử bật tắt, điều khiển thiết bị...



## TÀI LIỆU THAM KHẢO

- [1] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks
- [2] Florian Schroff, Dmitry Kalenichenko, James Philbin. Face Net: A Unified Embedding for Face Recognition and Clustering
- [3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich. Going deeper with convolutions
- [4] Nhận diện khuôn mặt với mạng MTCNN và FaceNet (Phần 1)  
<https://viblo.asia/p/nhan-dien-khuon-mat-voi-mang-mtcnn-va-facenet-phan-1-Qbq5QDN4ID8>
- [5] Support Vector Machine  
<https://machinelearningcoban.com/2017/04/09/smv/>
- [6] Mai Văn Hà, Nguyễn Thế Xuân Ly. Ứng dụng thuật toán Facenet xây dựng hệ thống nhận dạng khuôn mặt, Tạp chí khoa học và công nghệ - Đại học Đà Nẵng, 15/7/2021
- [7] Nhà thông minh là gì? Kiến thức tổng quan về Smart Home  
<https://cafeland.vn/kien-thuc/nha-thong-minh-la-gi-kien-thuc-tong-quan-ve-smart-home-107299.html>
- [8] Top 10 thương hiệu nhà thông minh ở Việt Nam  
<https://hunonic.com/nha-thong-minh-o-viet-nam/>
- [9] Firebase là gì? Ưu nhược điểm và các dịch vụ của Firebase  
<https://bkhost.vn/blog/firebase/>
- [10] Introduction to Firebase Realtime Database  
<https://www.raywenderlich.com/books/saving-data-on-android/v1.0/chapters/12-introduction-to-firebase-realtime-database>
- [11] User Authentication and CRUD Operation with Firebase Realtime Database in Android
- [12] HTTP là gì? Những ưu điểm khi sử dụng giao thức HTTP  
<https://bizflycloud.vn/tin-tuc/http-20-la-gi-loi-ich-va-chuc-nang-20180623103502001.htm>
- [13] Tổng quan về giao thức HTTP, phương thức truyền tải dữ liệu giữa Server – Client  
[https://tapit.vn/http-request-va-http-response-phuong-thuc-giao-tiep-giua-server-client/#:~:text=HTTP%20\(Hyper%20Text%20Transfer%20Protocol,response%20gi%E1%BB%AFa%20c%C3%A1c%20h%E1%BB%87%20th%E1%BB%91ng](https://tapit.vn/http-request-va-http-response-phuong-thuc-giao-tiep-giua-server-client/#:~:text=HTTP%20(Hyper%20Text%20Transfer%20Protocol,response%20gi%E1%BB%AFa%20c%C3%A1c%20h%E1%BB%87%20th%E1%BB%91ng)
- [14] Giao thức MQTT: Ưu điểm và ứng dụng  
<https://tapit.vn/giao-thuc-mqtt-uu-diem-va-ung-dung/#:~:text=%E2%80%93MQTT%20l%C3%A0%20m%E1%BB%99t>

[%20%E2%80%9Clight%20weight,v%E1%BB%81%20b%C4%83ng%20th%C3%B4ng%20%C4%91%C6%B0%E1%BB%9Dng%20truy%E1%BB%81n.](#)

[15] ATMEGA2560 Datasheet (PDF) – ATMEL Corporation

[16] Temperature and Humidity Module DHT11 Product Manual – AOSONG  
[https://components101.com/sites/default/files/component\\_datasheet/DHT11-Temperature-Sensor.pdf](https://components101.com/sites/default/files/component_datasheet/DHT11-Temperature-Sensor.pdf)

[17] Relay Datasheet – SONGLE RELAY  
[https://components101.com/sites/default/files/component\\_datasheet/12v-relay.pdf](https://components101.com/sites/default/files/component_datasheet/12v-relay.pdf)

[18] TSOP 4438 Datasheet, Equivalent, IR Receiver (PDF)  
<https://datasheetspdf.com/pdf/1105513/Vishay/TSOP4438/1>

[19] MQ6 SENSOR Datasheet (PDF)  
<https://datasheetspdf.com/datasheet/MQ6.html>

[20] Rain drop Sensor Module Datasheet  
<https://components101.com/sensors/rain-drop-sensor-module>

[21] SIM800L Datasheet, Equivalent, Hardware Design (PDF)  
<https://datasheetspdf.com/pdf/989664/SIMCom/SIM800L/1>

[22] LM2596 Datasheet (HTML) – List of Unclassified Manufacturers  
<https://www.alldatasheet.com/datasheet-pdf/pdf/134372/ETC1/LM2596.html>

## PHỤ LỤC

### A1. Quản lý các file.xml ban đầu trước khi xây dựng giao diện App Android

Các file.xml ban đầu trước khi xây dựng giao diện App Android bao gồm file về màu sắc, file về các chuỗi hiển thị, file về kiểu hiển thị cho các View. Các file này là bước thiết lập ban đầu, khi tới các công đoạn xây dựng về sau sẽ không cần lặp lại các bước giống nhau mà chỉ cần sử dụng các đoạn code trong các file này.

Để quản lý các file này, ta thực hiện theo đường dẫn: App – res – values. Trong thư mục values này gồm có 3 file.xml là: colors.xml, strings.xml và styles.xml tương ứng với các file.xml được giới thiệu ở trên.

#### a. File colors.xml

Chứa đựng các màu sắc sẽ được sử dụng tới trong quá trình xây dựng.

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>

    <color name = "red">#F80317</color>
    <color name = "orange">#FF5722</color>
    <color name = "yellow">#FFEB3B</color>
    <color name = "green">#33AE38</color>
    <color name = "blue">#1980D3</color>
    <color name = "darkBlue">#101091</color>
    <color name = "purple">#9C27B0</color>
    <color name = "black">#FF000000</color>
    <color name = "white">#FFFFFFF</color>
    <color name = "background">#0091FF</color>
    <color name = "colorPrimaryVariant">#1275CA</color>
    <color name = "colorPrimary">#1B38DA</color>
    <color name = "trans">#00000000</color>
</resources>
```

#### b. File strings.xml

Chứa đựng các chuỗi kí tự sẽ được hiển thị trên giao diện App Android.

```
<resources>
    <string name="app_name">QaSHome</string>

    <string name="sign_up">sign up for Smart Home App</string>
    <string name="enter_username">Enter your User Name</string>
    <string name="enter_password">Enter your Password</string>
    <string name="log_in">Log In</string>
    <string name="forgot_password">Forgot Password?</string>
    <string name="SQAHome">SQAHome</string>
    <string name="Wellcome_back">Wellcome Back</string>
    <string name="log_out">Log Out</string>
    <string name="life_condition">Life Condition</string>
    <string name="manage_your_house">Manage your house</string>
    <string name="livingroom">Livingroom</string>
    <string name="bedroom1">Bedroom 1</string>
    <string name="bedroom2">Bedroom 2</string>
    <string name="kitchen">Kitchen</string>
```

```

<string name="bathroom">Bathroom</string>
<string name="skylight">Skylight</string>
<string name="your_condition_life">Your life condition</string>
<string name="back">Back</string>
<string name="hust">Bách Khoa Hà Nội</string>
<string name="description">Description</string>
<string name="temperature">Temperature</string>
<string name="feel_like">Temp Like</string>
<string name="temp_min">Min Temp</string>
<string name="temp_max">Max Temp</string>
<string name="location">Location</string>
<string name="humid">Humidity</string>
<string name="wind_speed">Wind speed</string>
<string name="sun_rise">Sun rise</string>
<string name="sun_set">Sun set</string>
<string name="your_livingroom">Your Livingroom</string>
<string name="fan_off">FAN OFF</string>
<string name="fan_on">FAN ON</string>
<string name="update">UPDATE</string>
<string name="your_kitchen">Your Kitchen</string>
<string name="gas_concentration">Gas Concent</string>
<string name="update_condition">Update Life Condition</string>
<string name="your_bedroom1">Your Bedroom 1</string>
<string name="your_bedroom2">Your Bedroom 2</string>
<string name="your_bathroom">Your Bathroom</string>
<string name="your_skylight">Your Skylight</string>
<string name="mode">MODE</string>
<string name="manual">MANUAL</string>
<string name="automatic">AUTOMATIC</string>
<string name="open">OPEN</string>
<string name="close">CLOSE</string>
<string name="rain_status">Rain status</string>
<string name="curtain">Curtain</string>

</resources>

```

### c. File styles.xml

Chứa định các kiểu cơ bản khi thiết lập properties cho các View.

```

<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name = "TextView">
        <item name = "android:layout_width">wrap_content</item>
        <item name = "android:layout_height">wrap_content</item>
        <item name = "android:textColor">@color/white</item>
        <item name = "android:textSize">20sp</item>
        <item name = "android:textAllCaps">false</item>
        <item name = "android:textStyle">bold</item>
    </style>
    <style name = "EditTextView">
        <item name = "android:background">@color/white</item>
        <item name = "android:textColor">@color/black</item>
    </style>
</resources>

```

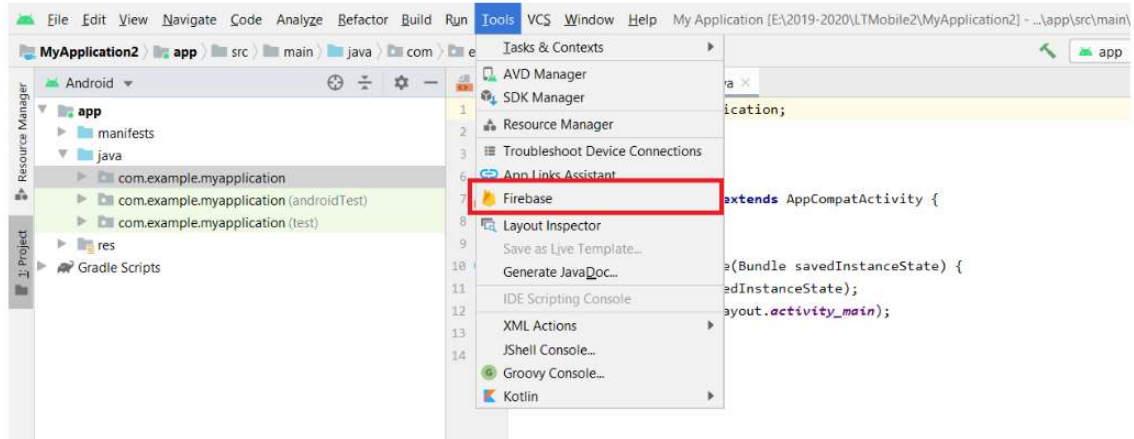
## A2. Kết nối Android Studio với Firebase

Trước tiên, cần phải có tài khoản Firebase với một project đã được tạo trước:

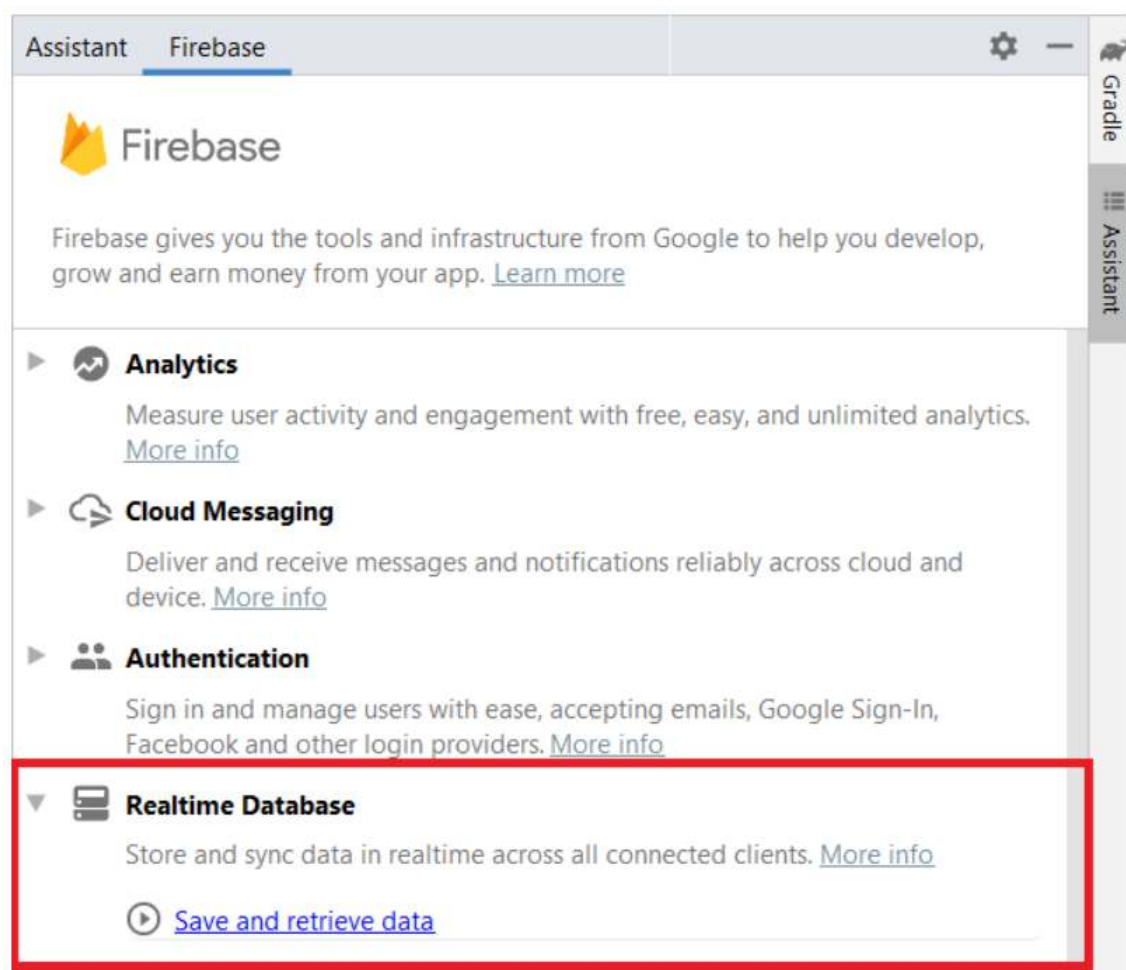
QASHproject  
qashproject-761b3



Trên thanh công cụ của Android Studio, chọn: **Tools – Firebase**



Trong hộp thoại Firebase, lựa chọn **Realtime Database – Save and retrieve data**



Hộp thoại **Save and retrieve data** hiển thị, tại đây sẽ hiển thị các bước để kết nối Android Studio với Firebase.

**Bước 1. Connect your app to Firebase**

① Connect your app to Firebase

Connect to Firebase



Thực hiện kết nối app Android và Firebase, hộp thoại hiển thị việc kết nối thành công như sau. Khi kết nối thành công tiếp tục thực hiện bước 2.

## ① Connect your app to Firebase

✓ Connected

### ***Bước 2. Add the Realtime Database to your app***

## ② Add the Realtime Database to your app

Add the Realtime Database to your app

Thực hiện đưa cơ sở dữ liệu thời gian thực vào ứng dụng Android. Các bước tiếp sau trong hộp thoại **Save and retrieve data** là các hướng dẫn trao đổi dữ liệu giữa Android Studio và Firebase Database đã được nhóm giới thiệu ở các mục trước đó.

### **A3. Thiết bị sử dụng**

Webcam Hikvision DS-U02 full HD 1080P/mic

**Samsung galaxy A20**

**HP ZBook 15 G2 CPU-i7 4800MQ**

### **A4. Các thư viện sử dụng**

Python 3.7.9:

pytsx3: thư viện hỗ trợ xử lý chuyển đổi văn bản sang âm thanh

speechrecognition: thư viện hỗ trợ nhận diện giọng nói

youtube-search: thư viện tìm kiếm video trên Youtube

playsound: thư viện âm thanh

gTTS: thư viện hỗ trợ chuyển đổi giữa các ngôn ngữ

Wikipedia: thư viện tra cứu thông tin trên Wikipedia

Google: thư viện tra cứu google

pipwin

pyaudio: thư viện âm thanh

cryptography: thư viện hỗ trợ mã hóa

pyserial: thư viện hỗ trợ giao tiếp cổng Com

tensorflow==1.15.5: nền tảng hỗ trợ machine learning

keras==2.3.1

scipy==1.1.0

scikit-learn

opencv-python: thư viện hỗ trợ xử lý ảnh

h5py

matplotlib: thư viện hỗ trợ các biểu đồ

Pillow: thư viện hỗ trợ xử lý ảnh cho trình biên dịch

requests

psutil

imutils

### **A5. Code Arduino điều khiển**

```
#include <Servo.h>
```

```
#include <DHT.h>
```

```
#include <string.h>
```

```
#include <SoftwareSerial.h> (Sim800L)
```

```
//Khai báo thông tin Living room
```

```
const int DHTPIN_living = 24;          //chân cảm biến DHT11 Livingroom là  
chân 24
```

```
const int DHTTYPE = DHT11;
```

```
DHT dht_living(DHTPIN_living, DHTTYPE);
```

```
Servo servo_left;                      //chân Servo bên trái là chân 2
```

```
Servo servo_right;                    //chân Servo bên phải là chân 5
```

```
int left_open;
```

```
int left_close;
```

```
int right_open;
```

```
int right_close;
```

```
int led_living = 10;                   //chân Led Living là chân 10
```

```
int fan_living = 4;                   //chân Fan Living là chân 4
```

```
//Khai báo thông tin Stair
```

```
int infrared = 28;                    //chân cảm biến hồng ngoại là chân 28
```

```
int led_stair = 26;                   //chân Led Stair là chân 26
```

```
//Khai báo thông tin Kitchen
```

```
const int DHTPIN_kitchen = 32;        //chân cảm biến DHT11 Kitchen là chân  
32
```

```
DHT dht_kitchen(DHTPIN_kitchen, DHTTYPE);
```

```
int gas_sensor = 52;                  //chân cảm biến khí gas Kitchen là chân 52
```

```

int gas_current;           //lưu trữ giá trị khí gas thời điểm đo
int led_kitchen = 50;      //chân Led Kitchen là chân 50;

Servo servo_kit;          //chân Servo Kitchen là chân 46
int kit_open = 90;
int kit_close = 0;

//Khai báo thông tin Bedroom1
const int DHTPIN_bedroom1 = 12;      //chân cảm biến DHT11 Bedroom1 là chân 12
DHT dht_bedroom1(DHTPIN_bedroom1, DHTTYPE);
int led_bedroom1 = 36;      //chân Led Bedroom1 là chân 36
int fan_bedroom1 = 11;      //chân Fan Bedroom1 là chân 11

//Khai báo thông tin Bedroom2
const int DHTPIN_bedroom2 = 48;      //chân cảm biến DHT11 Bedroom2 là chân 48
DHT dht_bedroom2(DHTPIN_bedroom2, DHTTYPE);
int led_bedroom2 = 44;      //chân Led Bedroom2 là chân 44
int fan_bedroom2 = 38;      //chân Fan Bedroom2 là chân 38

//Bathroom
int led_bathroom = 7;      //chân led của Bathroom là chân 7

//Khai báo thông tin Skylight
int rainSensor = A0;      //chân rain sensor là chân A0
int thresholdRain = 500;  //ngưỡng giá trị mưa là 500
const int limitswitch1 = 49;      //chân công tắc hành trình 1 à chân 49
const int limitswitch2 = 51;      //chân công tắc hành trình 2 à chân 51
Servo servo_rain;          //chân Servo bên trái là chân 37

//Khai báo chân giao tiếp giữa Arduino và Sim800L
SoftwareSerial sim(42, 40);      //tx-42, rx-40
String number = "0364887143";    //SĐT được gửi đến là 0364887143

String input_data;            //input_data nhận từ python xuống
String output_data;          //out_putdata gửi lên python

```

```

void setup() {
  Serial.begin(115200);

  //Livingroom
  servo_left.attach(2);
  servo_right.attach(5);
  dht_living.begin();
  pinMode(led_living, OUTPUT);
  pinMode(fan_living, OUTPUT);

  //Stair
  pinMode(led_stair, OUTPUT);
  pinMode(infrared, INPUT_PULLUP);

  //Kitchen
  dht_kitchen.begin();
  pinMode(gas_sensor, INPUT);
  pinMode(led_kitchen, OUTPUT);
  servo_kit.attach(46);

  //Bedroom1
  dht_bedroom1.begin();
  pinMode(led_bedroom1, OUTPUT);
  pinMode(fan_bedroom1, OUTPUT);

  //Bedroom2
  dht_bedroom2.begin();
  pinMode(led_bedroom2, OUTPUT);
  pinMode(fan_bedroom2, OUTPUT);

  //Skylight
  pinMode(rainSensor, INPUT);
  pinMode(limitswitch1, INPUT);
  pinMode(limitswitch2, INPUT);
}

void loop() {

```

```

    input_data = Serial.readString();           //đọc dữ liệu từ python là 1 chuỗi
    kí tự là: py|.|.|.|.|.
    output_data = "";
    if(input_data[0] == 'p' and input_data[1] == 'y'){    //hai kí tự đầu trong dãy kí
    tự điều khiển phải là "py"
                                                    //reset lại dữ liệu gửi lên python, chuỗi kí tự
    đó là: .....|.....|.....|.....
//Living Room
    if(input_data[2] == '0'){servo_living(0);}           //kí tự đọc thứ 3 là điều khiển
    2 servo_livingroom
    else {servo_living(1);}

    if(input_data[3] == '0'){fan_control(fan_living,0);} //kí tự đọc thứ 4 là điều
    khiển fan_livingroom
    else {fan_control(fan_living,1);}

    if(input_data[4] == '0'){led_control(led_living,0);} //kí tự đọc thứ 5 là điều
    khiển led_livingroom
    else {led_control(led_living,1);}

//-----

//Stair
    ir_control();

//-----

//Kitchen
    if(input_data[5] == '0'){servo_kitchen(0);}           //kí tự thứ 6 là điều khiển
    servo kitchen
    else {servo_kitchen(1);}

    if(input_data[6] == '0'){led_control(led_kitchen, 0);} //kí tự thứ 7 là điều khiển
    led kitchen
    else {led_control(led_kitchen, 1);}

//-----

//Bedroom1

```

```

    if(input_data[7] == '0'){fan_control(fan_bedroom1, 0);} //kí tự thứ 8 là điều
    khiển fan bedroom1
    else{fan_control(fan_bedroom1, 1);}

    if(input_data[8] == '0'){led_control(led_bedroom1, 0);} //kí tự thứ 9 là điều
    khiển led bedroom1
    else{led_control(led_bedroom1, 1);}

//-----

//Bedroom2
    if(input_data[9] == '0'){fan_control(fan_bedroom2, 0);} //kí tự thứ 10 là điều
    khiển fan bedroom2
    else{fan_control(fan_bedroom2, 1);}

    if(input_data[10] == '0'){led_control(led_bedroom2, 0);} //kí tự thứ 11 là điều
    khiển led bedroom2
    else{led_control(led_bedroom2, 1);}

//-----

//Skylight
    if(input_data[11] == '0'){skylight_control(0);} //kí tự 12 là điều khiển skylight
    else{skylight_control(1);}
}

output_data = "";
    output_data += dht_read(dht_living);           //kí tự gửi thứ 123 và 456 biểu
    diễn nhiệt độ và độ ẩm của Livingroom

    output_data += gas_read();                     //kí tự gửi thứ 789 biểu diễn nồng độ
    khí gas (0xx(%))

    output_data += dht_read_kitchen(dht_kitchen); //kí tự 101112 biểu diễn
    nhiệt độ kitchen

    output_data += dht_read(dht_bedroom1);         //kí tự 131415 biểu diễn độ
    ẩm bedroom1, kí tự 161718 biểu diễn nhiệt độ bedroom1

```

```
output_data += dht_read(dht_bedroom2);          //kí tự 192021 biểu diễn độ
ấm bedroom2, kí tự 222324 biểu diễn nhiệt độ bedroom2
```

```
output_data += rain_read();                      //kí tự 25 biểu diễn trạng thái trời
mưa (1,0)
```

```
output_data = "ad" + output_data;
Serial.println(output_data);
}
```

```
String dht_read(DHT dht){                      //Hàm đọc dữ liệu của cảm biến ở
các phòng
```

```
String out="";
float h = dht.readHumidity();
float t = dht.readTemperature();
if(h != 0){
    if(h < 10){out = ('0'+String(int(h*10)));}
    else {out += String(int(h*10));}
}
else{out == "000";}
```

```
if(t != 0){
    if(t < 10){out += ('0'+String(int(t*10)));}
    else {out += String(int(t*10));}
}
else{out += "000";}
```

```
return out;                                     //Trả về là chuỗi kí tự 6 số trong đó: ba số
đầu biểu diễn nhiệt độ, ba số sau biểu diễn độ ấm của phòng
}
```

```
void servo_living (int status){                //Hàm điều khiển Servo ở cửa
Livingroom
```

```
if(status == 0){
    left_close = 180;
    right_close = 0;
    servo_left.write(left_close);
    servo_right.write(right_close);
}
```



```

else
{
    left_open = 90;
    right_open = 90;
    servo_left.write(left_open);
    servo_right.write(right_open);
}
}

void led_control(int led, int status){           //Hàm điều khiển led ở các phòng
    if(status == 1){
        digitalWrite(led, HIGH);
    }
    else{
        digitalWrite(led, LOW);
    }
}

void fan_control(int fan, int status){           //Hàm điều khiển fan ở các phòng,
biến fan là chân Arduino nối với relay
    if(status == 0){
        digitalWrite(fan, HIGH);
    }
    else if(status == 1){
        digitalWrite(fan, LOW);
    }
}

void ir_control(){                               //Hàm điều khiển led_stair và IR
    if(digitalRead(infrared) == 0){
        digitalWrite(led_stair, HIGH);
    }
    else{
        digitalWrite(led_stair, LOW);
    }
}

```

```

void servo_kitchen (int status){ //Hàm điều khiển Servo ở cửa sổ
Kitchen
    if(status == 1){
        kit_open = 90;
        servo_kit.write(kit_open);
    }
    else{
        kit_close = 0;
        servo_kit.write(kit_close);
    }
}

```

```

String gas_read (){ //Hàm đọc dữ liệu cảm biến khí gas
    String out;
    gas_current = analogRead(gas_sensor);
    gas_current = (gas_current/1023)*100;
    if(gas_current == 0) return "000";
    if(gas_current < 10){out += ('0'+String(int(gas_current*10)))};
    else {out += String(int(gas_current*10));}

    return out; //Trả về là chuỗi kí tự 3 số biểu diễn nồng
    độ khí gas
}

```

```

String dht_read_kitchen(DHT dht){ //Hàm chỉ đọc nhiệt độ ở phòng
bếp
    String out;

    float t = dht.readTemperature();
    if(t < 10){out += ('0'+String(int(t*10)))};
    else {out += String(int(t*10));}

    return out; //Trả về là chuỗi kí tự 3 số biểu diễn nhiệt
    độ của phòng bếp
}

```

```

String rain_read(){ //Hàm đọc về thông tin trời mưa hay
không
    String out;

```

```

int rain_sensor = analogRead(rainSensor);
if(rain_sensor < thresholdRain){
    out += "1";
}
else if(rain_sensor >= thresholdRain){ out += "0";}
return out; //Trả về chuỗi kí tự 1 số biểu diễn trạng thái
mưa (1: mưa, 0: không mưa)
}

```

```

void sendMessage() { //Hàm thực hiện nhắn tin cảnh báo
    sim.println("AT+CMGF=1"); //Set up module sim dạng text
    sim.println("AT+CMGS=\"" + number + "\"\r"); //Gửi tin nhắn tới SĐT
    number
    String SMS = "Warning Gas!";
    sim.println(SMS);
    sim.println((char)26); //SMS dạng ASCII
}

```

```

void gasWarning() { //Hàm cảnh báo nếu nồng độ khí gas
    vượt quá 10% thì nhắn tin về cho người dùng
    gas_current = analogRead(gas_sensor);
    gas_current = (gas_current/1023)*100;
    if(gas_current > 10){
        sendMessage();
    }
}

```

```

void skylight_control(int status){
    if(status == 0){
        servo_rain.write(90);
    }
    else{
        servo_rain.write(0);
    }
}

```